Book Introduction by the Authors

Invited by

Kazuo Iwama

iwama@kuis.kyoto-u.ac.jp
Bulletin Editor
Kyoto University, Japan

BEATCS no 113

BOOLEAN FUNCTION COMPLEXITY

ADVANCES AND FRONTIERS

Stasys Jukna*

> Go to the roots of calculations! Group the operations. Classify them according to their complexities rather than their appearances! This, I believe, is the mission of future mathematicians.
>
> – Evariste Galois

What it is all about?

My book [5] is all about proving lower bounds.

Roughly speaking, research in Computational Complexity has two tightly interconnected strands. One of these strands—*structural complexity*—deals with high-level complexity questions: is space a more powerful resource than time? Does randomness enhance the power of efficient computation? Is it easier to verify a proof than to construct one? So far we do not know the answers to any of these questions; thus most results in structural complexity are *conditional* results that rely on various unproven assumptions.

My book [5] is about the life on the second strand—*circuit complexity*. Inhabitants of this strand deal with establishing *unconditional* lower bounds on the computational complexity of specific problems, like multiplication of matrices or detecting large cliques in graphs. This is essentially a low-level study of computation; it typically centers around particular models of computation such as decision trees, branching programs, boolean formulas, various classes of boolean circuits, communication protocols, proof systems and the like.

Why yet another book?

More than twenty years have passed since the well-known books on circuit complexity by Savage (1976), Nigmatullin (1983), Wegener (1987) and Dunne (1988)

*Goethe University Frankfurt, Germany, and Institute of Mathematics and Informatics, Vilnius University, Lithuania. jukna@thi.informatik.uni-frankfurt.de. Research supported by the DFG grant SCHN 503/6-1.

as well as a famous survey paper of Boppana and Sipser (1990) were written. Albeit in the meanwhile some excellent books in computational complexity appeared—including those by Savage (1998), Goldreich (2008) and Arora and Barak (2009)—these were mainly about the life on the first strand—structural complexity. So, it was the time to collect the new developments in circuit complexity during these two decades.

Almost everything is complex

It is known for now more than 70 years that most boolean functions require circuits of exponential size. In particular, Shannon, Lupanov and his students even established the following tight asymptotic for the maximum $\{\wedge, \vee, \neg\}$-circuit complexity $\mathrm{C}(n) = \max_f \mathrm{C}(f)$ of a boolean function of n variables:

$$1 + \frac{\log_2 n}{n} - O\Big(\frac{1}{n}\Big) \le \mathrm{C}(n) \cdot \frac{n}{2^n} \le 1 + \frac{\log_2 n}{n} + \frac{\log_2 \log_2 n}{n} + O\Big(\frac{1}{n}\Big)$$

Using these estimates, one can, say, easily prove the "Circuit Hierarchy Theorem": for every $n \le t(n) \le 2^{n-2}/n$, there are boolean functions computable by circuits of size $4t$, but having no circuits of size t. In a similar vein is the result that, for every $k \ge 1$, there exist boolean functions f_n of DNF-size n^{2k+1} such that $\mathrm{C}(f_n) > n^k$.

Unfortunately, these (and many other) results only show a mere *existence* of hard boolean functions. An ultimate goal of circuit complexity, however, is to exhibit such hard functions, and to understand *why* they are hard. Say, why the threshold function (does a given graph have k edges) is "simple", whereas the clique function (does a given graph has a clique with k edges) is "hard". And here, as in many other fields of mathematics—where the question comes to *construct* particular objects—the situation is much worse: the strongest known lower bounds on the unrestricted $\{\wedge, \vee, \neg\}$-circuit complexity of explicit boolean functions remain of the form cn for some small constants c; the current record remains $c = 5$.

Strong (even exponential) lower bounds were only obtained for various restricted circuit models. Below I give a rough overview of the book's contents.

Forget what was done: Formulas

Formulas are $\{\wedge, \vee, \neg\}$-circuits whose underlying graphs are trees. That is, these are the circuits without any memory: if we want to use some already computed (by a sub-formula) function g in another place, we are forced to re-compute g again. Some results:

- Formulas can be balanced: if f can be computed by a formula of size $\mathrm{L}(f)$, then f can be computed by a formula of depth $\mathrm{D}(f) \le 1.73 \log_2 \mathrm{L}(f)$. For

circuits, we only know that if f can be computed by a circuit of size S, then f can be also computed by a circuit of depth $O(S/\log S)$.

- The maximum of $D(f)$ over all boolean functions f of n variables is asymptotically equal $n - \log_2 \log_2 n$.

- If a boolean function f can be computed by a depth-d $\{\wedge, \vee, \neg\}$-formula using *unbounded* fanin AND and OR gates and having S leaves, then $D(f) \leq d - 1 + \lceil \log_2 S \rceil$. Note that a trivial upper bound, obtained by simulating each gate by a binary tree, is only $D(f) = O(d \log S)$.

- The depth of a circuit (or formula) is equal to the communication complexity of the following "find a separating bit" gate: Alice gets a vector $a \in f^{-1}(1)$, Bob gets a vector $b \in f^{-1}(0)$, and their goal is to find a bit $i \in [n]$ such that $a_i \neq b_i$.

- Khrapchenko's lower bound: Form a bipartite graph G_f with parts $f^{-1}(1)$ and $f^{-1}(0)$ by drawing an edge (a, b) if and only if a and b differ in exactly one bit. Then $L(f)$ is at least the product of the average degrees of the left and right parts of the graph G_f. This gives the lower bound $L(\oplus_n) \geq n^2$ for the Parity function $\oplus_n(x) = x_1 \oplus x_2 \oplus \cdots \oplus x_n$.

- There were many attempts to extend Khrapchenko's measure to obtain larger lower bounds. His measure is sub-modular and convex. It turned out, however, that no sub-modular or convex complexity measures can break down this quadratic barrier.

- A weaker bound $L(\oplus_n) = \Omega(n^{3/2})$ was earlier proved by Subbotovskaya by inventing the method of *random restrictions*. Currently, this method is widely used, in particular, to prove lower bounds for constant-depth circuits and communication protocols.

- When properly applied, Subbotovskaya's approach yields up to $\Omega(n^{3-o(1)})$ lower bounds, and this is a current record for $\{\wedge, \vee, \neg\}$-formulas.

- Other lower-bounds arguments for formulas are known as well: the method of "universal" functions for formulas where all binary boolean functions are allowed as gates, the method based on graph entropy, and the relation of formula size with the affine dimension of graphs.

- Lower bounds for *monotone* formulas were obtained by using rank as well as communication complexity arguments. In particular, rank arguments give tight superpolynomial lower bounds $n^{\Theta(\log n)}$ for functions induced by Paley graphs, as well as tight lower bounds for monotone quadratic functions.

Communication complexity arguments yield lower bounds $n^{\Theta(\log n)}$ even for such "simple" boolean functions as the *s-t* connectivity function.

- Superpolynomial lower bounds $n^{\Theta(\log n)}$ were also obtained for so-called monotone *span programs*, a model which may be even exponentially more powerful than monotone formulas. A span program for a boolean function $f(x_1, \ldots, x_n)$ is a 0/1 matrix whose rows are labeled by literals (variables and their negations); one literal can label several rows. A program is monotone if there are no negated labels. When an input $a \in \{0,1\}^n$ arrives, all rows whose labels are inconsistent with a are removed, and input a is accepted if the remaining rows span the all-1 vector over $GF(2)$.

Forbid negations: Monotone circuits

These are circuits with fanin-2 AND and OR gates, but no NOT gates. Despite of its seeming "simplicity", this model resisted any attempts to prove larger than linear lower bounds.

- The situation changed in 1985-86 when Razborov came with his "method of approximations", and proved a super-polynomial lower bound for the clique function. After that some modifications and extensions of his method were suggested. Razborov approximated gates by monotone DNFs and used the Sunflower Lemma of Erdős and Rado to convert CNFs to DNFs.

- Later, Sipser's notion of "finite limits" and a monotone Switching Lemma have led to a symmetric version of Razborov's argument, where both DNFs and CNFs are used to approximate gates (a two-side approximation). This resulted into the following general lower bounds criterion: if a monotone boolean function has a monotone circuit of size t, then it is *t-approximable*.

  Being t-approximable mean that there exist integers $2 \le r, s \le n$, a monotone s-CNF $C(x)$, a monotone r-DNF $D(x)$, and a subset $I \subseteq [n]$ of size $|I| \le s-1$ such that $|C| \le t \cdot (r-1)^s$, $|D| \le t \cdot (r-1)^s$, and either $C \le f$ or $f \le D \vee \bigvee_{i \in I} x_i$ hold. Important here is that the s-CNF C has only $t \cdot (r-1)^s$ out of all possible $\binom{n}{s}$ clauses, and similarly for the r-DNF D.

- This criterion holds even when any monotone *real-valued* functions $g : \mathbb{R}^2 \to \mathbb{R}$ are allowed as gates, and enables one to obtain in a uniform way strong lower bounds for a full row of explicit boolean functions. Together with appropriate interpolation theorems, these bounds have also led to the first exponential lower bounds for the length of the cutting-plane proofs.

- Why should one care about monotone circuits? The point is that this model has a purely "practical" importance. Namely, lower bounds for such circuits imply the same lower bounds for (min, +)-circuits, and hence, for dynamic programming. In this respect, our knowledge about the power of monotone circuits remains unsatisfying. Say, we still cannot prove that the *s*-*t* connectivity or even the connectivity function require circuits of size $\Omega(n^3)$. Known dynamic programming algorithms give circuits of size $O(n^3)$ for both these functions.

- It is known that there are monotone boolean functions f (like the Perfect Matching function) that can be computed by non-monotone circuits of polynomial size, but any monotone circuit for them must have a super-polynomial number of gates. This rises a question about the role of NOT gates.

- A classical result of Markov implies that $M(n) = \lceil \log_2(n+1) \rceil$ NOT gates are enough to compute any boolean function of n variables.

- Fisher and other authors substantially improved this by showing that restricting the number of NOT gates to $M(n)$ can only increase the size of a circuit by only an additive factor of $O(n \log^2 n)$.

- It is also know that the Markov–Fisher bound can almost be reached: there are explicit monotone (multi-output) boolean functions f which have polynomial size circuits only if more than $M(n) - O(\log \log n)$ NOT gates are used.

Restrict the time: Bounded-depth circuits

Yet another possibility to "bind circuits hands" is to allow NOT gates as well as AND and OR gates of unbounded fanin, but to restrict the depth (parallel time) of the circuit. This model is known as AC^0-circuits ("alternating circuits of constant depth"), and is currently quite intensively investigated.

- AC^0 circuits were considered by many authors since early 80's. When dealing with them, two major techniques emerged: the depth-reduction method via appropriate versions of the Switching Lemma, as well as approximation by low-degree polynomials.

- The depth-reduction argument has led to a tight $2^{\Theta(n^{1/(d-1)})}$ lower bound on the size of depth-d circuits computing the Parity function. Moreover, this function cannot even be approximated by such circuits of polynomial size.

- The approximation by low-degree polynomials argument has led to an exponential lower bound $2^{\Omega(n^{1/2d})}$ for the Majority function, even if Parity functions are also allowed as gates. Similar lower bounds were also proved when instead of Mod-2 gates, arbitrary prime-modulo functions are allowed as gates.

- The case of arbitrary, including *composite* modulo gates remains open. The class of boolean functions computable by such circuits of polynomial size is usually denoted by ACC^0. Still, Williams (2011) has recently shown that $\mathrm{NEXP} \not\subseteq \mathrm{ACC}^0$. This wakes a hope that we will be able to expose a boolean function $f \notin \mathrm{ACC}^0$ lying in NP or even in P. Actually, the Majority function still remains as a possible candidate.

- Even AC^0-circuits of depth-3 are interesting: by the results of Valiant, any lower bound $2^{\phi(n)}$ with $\phi(n) \gg n/\log\log n$ would give an example of a boolean function which cannot be computed by a linear size (fanin-2) circuit of logarithmic depth; proving such a bound is now a more than 30 years old open problem, and no such bound is known even for $\{\oplus, 1\}$-circuits.

- Known lower bounds for depth-3 circuits are only of the form $2^{\Omega(\sqrt{n})}$, and can be obtained using so-called "finite limits" and quite simple combinatorics. If we require that the circuit must have parity gates (instead of OR gates) at the bottom (next to the inputs) level, then arguments of graph complexity allow us to prove even truly exponential lower bounds $2^{\Omega(n)}$. Unfortunately, Valiant's construction does not carry over such circuits.

- Motivated by neural networks, people have also considered circuits with threshold gates. A boolean function $f(x_1, \ldots, x_n)$ is a threshold function if there exist real numbers $w_0, w_1, \ldots, w_n$ such that for every $x \in \{0,1\}^n$, $f(x) = 1$ if and only if $w_1x_1 + \cdots + w_nx_n \geq w_0$. For unbounded-depth circuits with threshold functions as gates, only linear lower bounds are known. Even depth-3 is here not well understood. Exponential lower bounds are only known for depth-2 circuits.

Restrict the time, but allow omnipotent power

In *general* circuits, *arbitrary* boolean functions are allowed as gates. The *size* of such a circuit is defined as the total number of wires (rather than gates). Of course, then every single-output boolean function f of n variables can be computed by a circuit of size n: just take one gate—the function f itself. The problem, however, becomes nontrivial if instead of one function, we want to *simultaneously* compute m boolean functions $f_1, \ldots, f_m$ on the same set of n variables $x_1, \ldots, x_n$, that is,

to compute an (n,m)-*operator* $f : \{0,1\}^n \to \{0,1\}^m$. Note that in this case the phenomenon which causes complexity of circuits is *information transfer* instead of *information processing* as in the case of circuits computing a single function.

- It is clear that every (n,m)-operator can be computed using nm wires, even in depth 1. However, already circuits of depth 2 constitute a rather non-trivial model: any operator with $\omega(n^2/\log\log n)$ depth-2 wire complexity also cannot be computed by linear-size, logarithmic-depth boolean circuits of fanin 2.
- The strongest known lower bounds for depth-2 are of the form $\Theta(n^{3/2})$, and were proved for natural operators like the product of two $0/1$ $\sqrt{n} \times \sqrt{n}$ matrices over $GF(2)$. These bounds were proved using particular entropy arguments.
- A lot of work was done when trying to prove strong lower bounds for general depth-2 circuits computing *linear* operators $f_A(x) = Ax$ over $GF(2)$. Lower bounds for such operators are usually derived using appropriate algebraic arguments (matrix rigidity) as well as graph-theoretic arguments (various superconcentration properties of graphs).
- The strongest known lower bound for linear operators f_A in depth 2 is about $n \cdot \phi(n)^2$ where $\phi(n) = (\ln n)/(\ln\ln n)$. The lower bound is proved using superconcentration properties. Unfortunately, it is known that such arguments cannot yield larger than $n \ln^2 n/\ln\ln n$ lower bounds. Interestingly, the *upper* bounds for these operators are proved in the class of linear circuits, i.e. depth-2 circuits with only Parity gates. In fact, the question on whether non-linear gates can help to compute linear operators over $GF(2)$ remains widely open.

Allow only to branch and join: Branching programs

Decision trees constitute one of the "simplest" models of computation, and a lot of interesting results were proved for it. Just to mention some of them:

- P = NP∩co-NP holds for decision tree *depth*; this is proved using elementary combinatorics.
- P ≠ NP ∩ co-NP holds for decision tree *size*; this is proved using spectral arguments.
- The depth of decision trees is related to sensitivity and block-sensitivity of the computed functions, as well as to the degree of their representation as polynomials.

- Non-trivial depth lower bounds are also known when arbitrary real threshold functions (not just $x_i \geq 1$) are used as decision predicates. In particular, the Inner Product function requires depth $n/2$ even in this generalized model.

The model of branching programs (BP) is a generalization of decision trees: the underlying graph may now be an arbitrary acyclic graph (not just a tree). The size here is the number of edges.

- For unrestricted BPs the progress was rather minor: the strongest lower bounds remain $\Omega(n^2/\log^2 n)$ for deterministic, and $\Omega(n^{3/2}/\log n)$ for nondeterministic BPs, both proved more than 40 years ago by Nechiporuk.
- For *symmetric* boolean functions, Nechiporuk's argument cannot yield any super-linear lower bounds. Such bounds were proved using more subtle arguments by many authors around 1990.
- One of the most surprising results for general BPs is the theorem of Barrington stating that branching programs of width-5 are not much weaker than formulas.
- Exponential lower bounds for BPs were proved only when either each variable can be re-tested constant times along each computation path, or when at most cn variables for a sufficiently small constant $c > 0$ are allowed to be tested more than once along each computation. The arguments here use a rather non-trivial combinatorics, probabilistic arguments as well as expander graphs.
- Still, the situation even with restricted BPs remains rather unsatisfying. In particular, we are still unable to prove any strong lower bounds for the following one of the simplest non-deterministic models of "almost read-once" BPs: these are nodeterministic BPs where every consistent paths must be read-once (no variable can be tested more than once). The problem here is that we have no restrictions on inconsistent paths (those containing contradictory tests $x_i = 0$ and $x_i = 1$ on some variable).

Allow only to chat: Communication complexity

Since communication complexity has a comprehensive treatment in an excellent book by Kushilevitz and Nisan of 1997, we have restricted ourselves to results essentially used later in our book, as well to some newer results. In particular, we describe the progress concerning the so-call "rank-conjecture", prove that P = NP ∩ co-NP holds for fixed-partition games, whereas P ≠ NP ∩ co-NP holds for best-partition games, present lower bounds on randomized protocols, and Forster's (2002) celebrated lower bound on the sign-rank of ±1 matrices.

Applications: Proof complexity

The last two chapters of the book are devoted to some applications of the previous results when lower-bounding the length of resolution and cutting-planes proofs. The point is that so-called regular resolution proofs are, in fact, read-once branching programs solving particular search problems (find an unsatisfied clause in the given CNF). On the other hand, the length of cutting-plane proofs can be lower-bounded using some communication complexity arguments or using the interpolation theorem together with lower bounds on the size of monotone circuits with real-valued gates.

What's new: Some features

- The book discusses some topics, like graph complexity or method of finite limits, that are not known well enough even for specialists in circuit complexity.
- Gives new proofs of classical results, like lower bounds for monotone circuits, monotone span programs and constant-depth circuits.
- Presents some topics never touched in existing complexity books, like graph complexity, span programs, bounds on the number of NOT gates, bounds on Chvátal rank, lower bounds for circuits with arbitrary boolean functions as gates, etc.
- Relates the circuit complexity with one of the "hottest" nowadays topics – the proof complexity.
- Contains more than 40 specific open problems, two of which were already re-solved after the book was published.
- The main feature, however, is the inclusion of many results of Russian mathematicians which remained unknown in the West. Just to give an example, the following result proved by Lupanov already in 1956 was later re-discovered by many authors (with much more involved proofs): every bipartite $n \times m$ graph can be decomposed into edge-disjoint bipartite cliques so that the sum of their nodes does not exceed $(1 + o(1))nm/\log_2 n$.

Epilogue

At the end of the book, I shortly sketch some stuff not discussed in the main text: pseudo-random generators, natural proofs, the fusion method for proving

lower bounds, and indirect (diagonalization) arguments. The Appendix contains all necessary mathematics.

Acknowledgement

I am thankful to Sasha Razborov for his comments on this summary.

References

[1] S. Arora and B. Barak (2009): *Computational Complexity: A Modern Approach*. Cambridge University Press. www.cs.princeton.edu/theory/complexity/

[2] R. B. Boppana and M. Sipser (1990): The complexity of finite functions, in: Handbook of Theoretical Computer Science, Volume A: Algorithms and Complexity (A), 757-804.

[3] P. E. Dunne (1988): *The Complexity of Boolean Networks*. Academic Press Professional, Inc., San Diego, CA. http://cgi.csc.liv.ac.uk/~ped/RESUME.html

[4] O. Goldreich (2008): *Computational Complexity: A Conceptual Perspective*. Cambridge University Press. www.wisdom.weizmann.ac.il/~oded/cc.html

[5] S. Jukna (2012): *Boolean Function Complexity: Advances and Frontiers*. Springer-Verlag. The home page of the book with a suplementary material: www.thi.cs.uni-frankfurt.de/~jukna/boolean/

[6] R. G. Nigmatullin (1983): *The Complexity of Boolean Functions*. Izd. Kazansk. Univ. (Kazan University Press, in Russian).

[7] J. E. Savage (1976): *The Complexity of Computing*. Wiley, New York.

[8] J. E. Savage (1998): *Models of Computation: Exploring the Power of Computing*. Addison-Wesley. The book is freely available for download at:

http://www.cs.brown.edu/~jes/book/home.html

[9] I. Wegener (1987): *The Complexity of Boolean Functions*. Wiley-Teubner. The book is freely available for download at:

http://eccc.hpi-web.de/static/books/The_Complexity_of_Boolean_Functions/