Some Estimated Likelihoods For Computational Complexity

R. Ryan Williams

MIT CSAIL & EECS, Cambridge MA 02139, USA

Abstract. The editors of this LNCS volume asked me to speculate on open problems: out of the prominent conjectures in computational complexity, which of them might be true, and why?
I hope the reader is entertained.

1 Introduction

Computational complexity is considered to be a notoriously difficult subject. To its practitioners, there is a clear sense of *annoying* difficulty in complexity. Complexity theorists generally have many intuitions about what is "obviously" true. Everywhere we look, for every new complexity class that turns up, there's another conjectured lower bound separation, another evidently intractable problem, another apparent hardness with which we must learn to cope. We are surrounded by spectacular consequences of all these obviously true things, a sharp coherent world-view with a wonderfully broad theory of hardness and cryptography available to us, but — *gosh, it's so annoying!* — we don't have a clue about how we might prove any of these obviously true things. But we try anyway.

Much of the present cluelessness can be blamed on well-known "barriers" in complexity theory, such as relativization [BGS75], natural properties [RR97], and algebrization [AW09]. Informally, these are collections of theorems which demonstrate strongly how the popular and intuitive ways that many theorems were proved in the past are fundamentally too weak to prove the lower bounds of the future.

- Relativization and algebrization show that proof methods in complexity theory which are "invariant" under certain high-level modifications to the computational model (access to arbitrary oracles, or low-degree extensions thereof) are not "fine-grained enough" to distinguish (even) pairs of classes that seem to be obviously different, such as NEXP and BPP.
- Natural properties also show how the generality of many circuit complexity lower bound proofs can limit their scope: if a method of proving circuit complexity lower bounds applies equally well to proving lower bounds against *random* functions, then it had better not be a highly constructive argument, where one can feasibly discern the circuit complexity of a simple function. Otherwise, our method for proving circuit lower bounds also proves an upper

bound, showing that pseudorandom functions cannot be implemented in the circuit class.

In any case, these barriers show that many known proof methods have so much slack in their arguments that, when it comes to questions like P versus NP, the method will simply hang itself. To make further progress on complexity class separations and prove these obviously-true theorems, we need to dig into computation deeper, and find less superficial methods of argument which speak about computation on a finer level.

I think it is highly probable that a decent level of cluelessness is due to simply being *wrong* about some of these obviously true things. I'm certainly not the first to proclaim such an opinion; Lipton and Regan's blog and books [Lip10,LR13] have spoken at length about how everyone was wrong about X for all sorts of X, and other "contrarian" opinions about complexity can be found in Gasarch's polls on P vs NP [Gas02,Gas12]. The idea that complexity theorists can be very wrong is certainly not in doubt.[1] The fact that it happens at a non-trivial frequency is enough that (I think) folks should periodically reconsider the conjectured complexity separations they have pondered over the years, and update their thoughts on them as new information arises. Regardless of one's opinions about how wrong we may or may not be, I think it is an important exercise to review the major problems in one's field once a year, and seriously check if you got any smarter about them over the previous year.

Moreover, I claim that complexity theorists are more often wrong about their lower bound conjectures than their upper bound conjectures. (Two recent occurrences are the non-rigidity of Hadamard/Sylvester matrices [AW17] which had been conjectured for decades to be rigid, along with the construction of good linear codes that are also not rigid [Dvi17].) This observation is quite probably due to the extremely useful and natural (conservative) heuristic that:

> *If a bunch of smart people could not figure out how to do it, then it probably cannot be done.*

So, when no good upper bound (i.e., algorithm) is attained for a problem, even after a bunch of smart people have thought about it, the inclination is to conclude that the upper bound does not exist (i.e., a lower bound). Hence it is natural that beliefs about lower bounds tend to be refuted more often than those about upper bounds: we rarely assert that interesting upper bounds exist, unless we already know how to attain them. (An interesting exception is that of matrix multiplication over a field; researchers in that area tend to believe that nearly-optimal running time is possible for the problem.) There seems to be an additional belief in the justification of the above heuristic:

> *As the bunch of smart people who cannot find a good algorithm increases over time, we get closer to a universal quantifier over all good algorithms.*

[1] Just ask my students!

For example, our collective inability to find an efficient SAT algorithm, even over decades of thought about the problem, even though all the other thousands of NP-complete problems are *really only SAT in disguise*, suggests to many that $\mathsf{P} \neq \mathsf{NP}$.

Unfortunately, I do not believe that the "bunch of smart people" living in the present time covers this sort of quantifier well, and I am not sure how we will raise the next generation of smart people to cover it more thoroughly (other than teaching them to be skeptical, and providing them a vastly-thicker literature of algorithms and complexity than what we had). For this reason, I am probably less dogmatic than a "typical" complexity theorist regarding questions such as P versus NP.

1.1 Some Estimated Likelihoods for Some Major Open Problems

I decided to present my perspective on some well-known open problems in complexity theory with a table of my personal "estimated likelihood" values for each one. Here is the table:

Proposition	**RW's Estimated Likelihood**
TRUE	100%
$\mathsf{EXP}^{\mathsf{NP}} \neq \mathsf{BPP}$	99%
$\mathsf{NEXP} \not\subset \mathsf{P/poly}$	97%
$\mathsf{L} \neq \mathsf{NP}$	95%
$\mathsf{NP} \not\subset \mathsf{SIZE}(n^k)$	93%
$\mathsf{BPP} \subseteq \mathsf{SUBEXP}$	90%
$\mathsf{P} \neq \mathsf{PSPACE}$	90%
$\mathsf{P} \neq \mathsf{NP}$	80%
ETH	70%
$\mathsf{NC1} \neq \mathsf{TC0}$	50%
$\mathsf{NEXP} \neq \mathsf{EXP}$	45%
SETH	25%
$\mathsf{NEXP} \neq \mathsf{coNEXP}$	20%
NSETH	15%
$\mathsf{L} \neq \mathsf{RL}$	5%
FALSE	0%

Table 1. What you receive, when you ask for my opinions on some open problems in complexity theory.

The numerical values of my "estimated likelihoods" are (obviously) nothing too rigorous. What is more important is the relative measure between problems. I did want the measures to be "consistent" in some simple senses. For example, if we know that A implies B, then B should not be (much) less likely to be true than A. I will give some explanations for my likelihoods in the next section.

There are many other open problems for which I could have put a likelihood, but I wanted to focus on problems where I thought I had something interesting to say along with my measure of likelihood. I deliberately refrained from putting a measure on conjectures which I do not feel that I am very knowledgeable on the state-of-the-art, such as the famous Unique Games Conjecture of Khot [Kho02]. For the record, the present state of knowledge suggests to me that Unique Games (as usually stated) is probably *intractable*, but perhaps not NP-hard. But what do I know? A very recent line of work [KMS17,KMS18] claims to settle the 2-to-2 conjecture, a close relative of Unique Games.

2 Thoughts on Various Separations

I will discuss the separations mentioned in Table 1.1, starting with those that I think are most likely to be true.

2.1 EXP with an NP oracle versus BPP

Recall that BPP is the class of problems solvable in randomized polynomial time (with two-sided error), and $\mathsf{EXP}^{\mathsf{NP}}$ is the class of problems solvable in (deterministic) exponential time with access to an oracle for the SAT problem (note that exponentially-long SAT instances can be solved in one step with such a model). I put 99% on the likelihood of $\mathsf{EXP}^{\mathsf{NP}} \neq \mathsf{BPP}$, for several reasons. One reason is that everything we know indicates that randomized computation is *far, far* weaker than deterministic exponential time, and exponential time with an NP oracle should be only more powerful. Another reason is that the open problem becomes trivially closed (separating the two classes) if one makes various small changes in the problem statement. Change the "two-sided error" to "one-sided error" (the class RP) and it is easy to separate them. Change the EXP to ZPEXP (randomized exponential time with "zero-error") and it is again easy to separate them. For a third reason, $\mathsf{EXP}^{\mathsf{NP}} \neq \mathsf{BPP}$ is implied by very weak circuit lower bounds (such as $\mathsf{NEXP} \not\subset \mathsf{P/poly}$) that I am also very confident are true (they will be discussed later).

It appears to me that $\mathsf{EXP}^{\mathsf{NP}} \neq \mathsf{BPP}$ is primarily still open because there are oracles making them equal [Hel86], so one will need to use the right sort of non-relativizing argument to get the job done. I do not view the existence of oracles as a significant barrier, but rather a caution sign that we will need to dig into the guts of Turing machines (or equivalent formalizations) in order to separate the classes. Some potential approaches to (weak) lower bounds against BPP are outlined in an earlier article of mine [Wil13b].

2.2 NEXP vs P/poly

Recall that NEXP is the class of problems solvable in nondeterministic exponential time: a huge complexity class. The class P/poly is a special kind of class: it

consists of those problems over $\{0,1\}^\star$ which can be solved by an *infinite family* of polynomial-size circuits $\{C_n\}$. Intuitively, this is a computational model with an infinitely-long description (a so-called *non-uniform* model), but for any particular input length n, the description of the computer solving the problem on all inputs of length n (and the running time of this computer) is a fixed polynomial of n. I put 97% likelihood on $\mathsf{NEXP} \not\subset \mathsf{P/poly}$. Note that this lower bound would imply $\mathsf{EXP}^{\mathsf{NP}} \neq \mathsf{BPP}$.

I can think of two major reasons why this separation is almost certainly true.

2.2.1 Why Would Non-Uniformity Help Here? I can see no reason why the non-uniform circuit model should let us significantly speed-up the solution to every NEXP problem (or to every EXP problem, for that matter). Having a distinct algorithm for each input length does not intuitively seem to be very helpful: how could it be that every input length n allows for some special hyper-optimization on that particular n, yielding an exponentially faster solution to an NEXP problem? And how could this happen to be true for *every* individual length n? In this light, it feels remarkable to me that this separation problem is still open at all.

Note that there are still oracles relative to which NEXP is contained in P/poly, even in the algebrization sense [AW09]. It is known that there are *undecidable* problems in P/poly, but this is because we can concoct undecidable problems out of *unary* (or more generally, sparse) languages.

I am willing to entertain the possibility that there are *infinitely many* input lengths for which NEXP problems are easy: perhaps NEXP is contained *infinitely often* in P/poly. For example, the "good" input lengths could have the form $2^{2^{\cdot^{\cdot^{\cdot^{2}}}}}$, or something more bizarre. This would be an amazing result, but because we only require infinitely many input lengths to work, perhaps some hyper-optimization of certain strange input lengths is possible in this setting. There are oracles relative to which NEXP is contained infinitely often in NP [For15], which shows that infinitely-often simulations can be very tricky to rule out. Still, this also seems unlikely.

2.2.2 Extremely Weak Derandomization If the first reason was not already enough, the second reason for believing $\mathsf{NEXP} \not\subset \mathsf{P/poly}$ is that extremely weak derandomization results would already imply the result. More precisely, it is generally believed that $\mathsf{P} = \mathsf{BPP}$. A productive way to think about $\mathsf{P} = \mathsf{BPP}$ is to study a particular approximation problem, often called CAPP:

Circuit Approximation Probability Problem (CAPP)
Input: A Boolean circuit C with n inputs
Output: The quantity $\Pr_{x \in \{0,1\}^n}[C(x) = 1]$, to within $\pm\, 1/10$.

(Note the choice of $1/10$ is arbitrary, and could be any constant in $(0, 1/2)$.) It is known that a deterministic polynomial time algorithm for CAPP would imply $\mathsf{P} = \mathsf{BPP}$. From the work of Impagliazzo and Wigderson [IW97] on pseudorandom generators, such an algorithm follows from assuming that $\mathsf{TIME}[2^{O(n)}]$ does not (infinitely often) have $2^{\delta n}$-size circuits, for some $\delta > 0$. It was shown by Impagliazzo, Kabanets, and Wigderson [IKW02] that a deterministic $2^{n^{\varepsilon}}$-time algorithm for CAPP, for every $\varepsilon > 0$ would already imply $\mathsf{NEXP} \not\subset \mathsf{P/poly}$.[2] So, sub-exponential time deterministic algorithms for CAPP imply the NEXP circuit lower bound.

But in fact something even stronger can be said. For a circuit C with n inputs and size s, the brute-force algorithm for deciding CAPP takes no more than $2^n \cdot \mathsf{poly}(s)$ time. I showed [Wil10] that deciding CAPP in deterministic time

$$O(2^n \cdot \mathsf{poly}(s)/\alpha(n)),$$

for any super-polynomial function $\alpha(n)$, would already imply $\mathsf{NEXP} \not\subset \mathsf{P/poly}$. That is, *any* significant improvement over exhaustive search for CAPP would imply the lower bound we seek.[3] I strongly believe that such an algorithm exists, but it may be tough to find. Several circuit lower bounds against NEXP have indeed been proved by giving non-trivial SAT algorithms for various circuit classes [Wil11,Wil14,Tam16,ACW16,COS17].

2.3 LOGSPACE vs NP

I also believe that $\mathsf{L} \neq \mathsf{NP}$ is extremely likely: that (for example) the Vertex Cover problem on m-edge graphs cannot be solved by any algorithm that uses only m^k time (for some constant k) and $O(\log m)$ additional space (beyond the $O(m \log(n))$ bits of input that lists the edges of the graph). This is far from a controversial position; $O(\log m)$ space is not enough to even store a subset of nodes from the graph (a candidate vertex cover), and it is widely believed that this tiny space requirement is a severe restriction on computational power. In particular it is also widely believed that $\mathsf{L} \neq \mathsf{P}$ (which I would put less likelihood on, but not much less).

I mainly want to highlight $\mathsf{L} \neq \mathsf{NP}$ because (unlike the situation of $\mathsf{P} \neq \mathsf{NP}$, which is murkier) I believe that substantial progress has already been made on the problem. Significant combinatorial approaches to space lower bounds (such as [BJS98,Ajt99,BSSV00,BV02]) have yielded model-independent super-linear time lower bounds on decision problems in P, when the space usage is $n^{1-\varepsilon}$ or less. (In fact, these results hold in a non-uniform version of time-space

[2] In fact, a "nondeterministic" algorithm for CAPP of this form would be enough. I will refrain here from defining what such an algorithm means, and refer the reader to the paper [IKW02].

[3] Again, a "nondeterministic" CAPP algorithm with this property would already be enough.

bounded computation.) Approaches based on diagonalization/simulation methods, aimed at proving lower bounds on NP-hard decision problems such as SAT, include [For00,FLvMV05,Wil08a,Wil13a,BW12] and show that problems such as SAT, Vertex Cover, and Independent Set require $n^{2\cos(\pi/7)}$ time to be solved when the space usage of the algorithm is $n^{o(1)}$. Unfortunately, $2\cos(\pi/7) < 1.81$, and in fact the last reference above shows that current techniques cannot improve this curious exponent. So in a quantitative sense, we have a long way to go before $\mathsf{L} \neq \mathsf{NP}$.

After studying these methods for years now, I am more-or-less convinced of $\mathsf{L} \neq \mathsf{NP}$ and that it will be proved, possibly long before P vs NP is resolved. In fact I believe that only a few new ideas will be required to yield enough "bootstrapping" to separate L and NP. The catch is that I am afraid the missing ideas will need to be extraordinarily clever, unlike anything seen before (at least a Gödel-incompleteness-level of cleverness, relative to the age in which he proved those famous theorems). In the meantime, we do what we can.

2.4 NP Does Not Have Fixed Polynomial-Size Circuits

Recall that $\mathsf{SIZE}(n^k)$ is the class of problems solvable with Boolean circuits (of fan-in two) with $O(n^k)$ gates. Here we are investigating the likelihood of the proposition

$$\forall k \in \mathbb{N}, \mathsf{NP} \not\subset \mathsf{SIZE}(n^k).$$

That is, for every k, there is some problem in NP that doesn't have $O(n^k)$-size circuits.

First, I put a bit less likelihood (93%) on $\mathsf{NP} \not\subset \mathsf{SIZE}(n^k)$ for some constant k, because it implies $\mathsf{NEXP} \not\subset \mathsf{P/poly}$ (and they don't seem to be equivalent), so it is stronger than the other circuit lower bound problems that have been mentioned so far.

There is a considerable history of results in this direction. Kannan [Kan82] proved "fixed polynomial" circuit lower bounds for the class $\mathsf{NP}^{\mathsf{NP}}$ (a.k.a. $\Sigma_2\mathsf{P}$): for every constant $k \geq 1$, there is a problem in $\mathsf{NP}^{\mathsf{NP}}$ that does not have n^k size Boolean circuits (over any gate basis that you like). Over time, his fixed-polynomial lower bound has been improved several times, from $\mathsf{NP}^{\mathsf{NP}}$ to seemingly smaller complexity classes such as $\mathsf{ZPP}^{\mathsf{NP}}$ [KW98]. It is known that MA/1 (Merlin-Arthur with one bit of advice) is not in $\mathsf{SIZE}(n^k)$ for each k [San07], and due to our beliefs about circuit lower bounds [IW97] it is believed that $\mathsf{MA} = \mathsf{NP}$ (i.e., it is believed that randomness doesn't help much with non-interactive verification of proofs). This looks like strong evidence in favor of $\mathsf{NP} \not\subset \mathsf{SIZE}(n^k)$, besides the intuition that NP problems that require $n^{100000k}$ nondeterministic time probably can't be "compressed" to n^k-size circuits.

The problems of proving that classes such as P, NP, and P^{NP} have fixed polynomial-size circuits are discussed in [Lip94,FSW09,GM15,Din15], and many absurd-looking consequences have been derived from propositions such as $\mathsf{NP} \subset \mathsf{SIZE}(n^k)$ (of course, none of these have been proved to be actually contradictory).

Here's an example from [FSW09]. $\mathsf{P}^{\mathsf{NP}} \subset \mathsf{SIZE}(n^k)$ implies that for **every** NP verifier V, and every yes-instance x for the verifier V, there is a witness y_x that is extremely compressible: it can be represented by a circuit of only $O(|x|^k)$ size. To see this, note that the problem

> *Given an x and an integer i, print the ith bit of the lexicographically first y such that $V(x, y)$ accepts (or print 0 if no such y exists)*

is in P^{NP}, and therefore has $O(n^k)$-size circuits under the hypothesis. Thus the witnesses printed by these circuits have low circuit complexity, for every x.

2.5 BPP is in Sub-Exponential Time

Recall $\mathsf{SUBEXP} = \cap_{k \in \mathbb{N}} \mathsf{TIME}(2^{n^{1/k}})$, i.e., it is the class of problems solvable in $O(2^{n^{\varepsilon}})$ time, for any $\varepsilon > 0$ as close to zero as you like.

The main reason for putting a high likelihood on $\mathsf{BPP} \subseteq \mathsf{SUBEXP}$ is that it is implied by $\mathsf{EXP} \not\subset \mathit{io}\text{-}\mathsf{P/poly}$ [NW94,BFNW93], which I also believe to be true, although perhaps not quite as strongly as $\mathsf{NEXP} \not\subset \mathsf{P/poly}$. (The "io" part stands for "infinitely often" and it means that there is a function EXP which, for almost every input length n, fails to have circuits of size $\mathsf{poly}(n)$.) The intuition for why $\mathsf{EXP} \not\subset \mathit{io}\text{-}\mathsf{P/poly}$ is similar to the intuition for why $\mathsf{NEXP} \not\subset \mathsf{P/poly}$. As a starting point, one can easily prove results like $\mathsf{EXP} \not\subset \mathit{io}\text{-}\mathsf{TIME}[2^{n^k}]$ for every constant k, by diagonalization. It would be very surprising, even magical, if one could take a problem that cannot be solved infinitely-often in 2^{n^k} time and solve it infinitely-often in polynomial time, simply because one got a separate algorithm and received polynomially-long extra advice for each input length. Intuitively, to solve the hardest problems in EXP, the only advice that could truly help you solve the problem quickly (on all inputs of length n) would be the entire 2^n-bit truth table for length n, and for such problems it is not clear why some input lengths would ever be easier than others. (Aside: it is interesting to note that $\mathsf{P} = \mathsf{NP}$ implies circuit lower bounds such as $\mathsf{EXP} \not\subset \mathit{io}\text{-}\mathsf{P/poly}$.)

For what it's worth, I would put about 87% likelihood on $\mathsf{P} = \mathsf{BPP}$: a bit lower than the 90% BPP in sub-exponential time (for good reason), but not significantly less. These two likelihoods aren't substantially different for me, because I tend to believe that non-trivial derandomizations of BPP are likely to imply much more efficient derandomizations, along the lines of Theorems 1.4 and 1.5 in [Wil10].

2.6 P vs PSPACE

I have put a little less likelihood (90%) on $\mathsf{P} \neq \mathsf{PSPACE}$ than the previous lower bounds mentioned such as $\mathsf{L} \neq \mathsf{NP}$. I feel that less intellectual progress has been made on separating P from PSPACE, and so the extent to which we should believe

$\mathsf{P} \neq \mathsf{PSPACE}$ is less understood. For example, we don't know an $n^{1.0001}$ time lower bound against any PSPACE-hard problem solvable in linear space, such as quantified Boolean formula satisfiability (but we do know a few non-trivial-but-not-so-great lower bounds [HPV77,Wil08b,LW13]). The reminder that such a time lower bound is still open should signal a call-to-arms for complexity theorists:

> *If* PSPACE *is so large, why can't we prove an* $n^{1.0001}$ *time lower bound against solving QBF? What are the obstacles? Can we articulate some interesting tools that would suffice to prove the lower bound, if we had them?*

Nevertheless, $\mathsf{P} = \mathsf{PSPACE}$ *does look extremely unlikely*: the idea that PSPACE corresponds to computing winning strategies in two-player games makes it clear that our world would be extremely weird and wonderful if $\mathsf{P} = \mathsf{PSPACE}$ and we discovered a fast algorithm for solving quantified Boolean formulas.

2.7 P vs NP

I do not have much to say about P versus NP beyond what has already been said, over many decades, by many researchers (a notable example is Aaronson's astounding recent survey [Aar16]). But I do only give 80% likelihood of $\mathsf{P} \neq \mathsf{NP}$ being true. Why only 80%? Because the more I think about P versus NP, the less I understand about it, so why should I be so confident in its answer? Because ETH is less likely to be true than $\mathsf{P} \neq \mathsf{NP}$, and I feel like the truth of ETH is not so far from a coin toss. Because it is not hard, when you squint, to view incredible achievements like the PCP theorem [AS98,ALM+98] as progress towards $\mathsf{P} = \mathsf{NP}$ (one only has to satisfy $7/8+\varepsilon$ of the clauses in a MAX-3-SAT instance! [Hås01]) instead of hardness for approximately solving NP problems.

Yes, intuitively and obviously $\mathsf{P} \neq \mathsf{NP}$ — but only intuitively and obviously. (Incidentally, I put about the same likelihood on the existence of one-way functions; I tend to believe in strong worst-case to average-case reductions.)

2.8 ETH: The Exponential Time Hypothesis

Recall that ETH asserts that

> 3SAT on n variables cannot be solved in $2^{\varepsilon n}$ time, for some $\varepsilon > 0$.

I put only 70% likelihood on ETH. My chief reason (which will also appear later when I discuss NEXP and EXP) is that we simply do not yet have a somewhat-comprehensive understanding of what can be solved via sub-exponential time algorithms. That is not for a lack of trying: it is a very active subject (see for example [FK10]). Our understanding of polynomial-time algorithms is fairly

deep, but even there we have very few lower bounds: we know a lot less about what *cannot* be done.

Although I give it only 70% likelihood, I do not think it is necessarily presumptuous to base a research program on ETH being true, but I do think researchers should be a little more skeptical of ETH, and periodically think seriously about how it might be refuted. (As Russell Impagliazzo often reminds me: "It's not a Conjecture, it's a Hypothesis! We chose that word for a reason.") More points along these lines will be given when SETH (the Stronger ETH) is discussed.

2.9 NC1 versus TC0

Recall that NC^1 ("Nick's Class 1") is the class of problems solvable with $O(\log n)$-depth circuits of polynomial size and constant fan-in for each gate. The class TC^0 contains problems solvable with $O(1)$-depth fan-in circuits of polynomial size with unbounded fan-in MAJORITY gates along with inverters. (The T stands for "Threshold" — without loss of generality, the gates could be arbitrary linear threshold functions.) It is well-known that $\mathsf{TC}^0 \subseteq \mathsf{NC}^1$, and $\mathsf{NC}^1 \neq \mathsf{TC}^0$ is sometimes used as a working hypothesis.

Upon reflection, I have possibly put significantly less weight on $\mathsf{NC}^1 \neq \mathsf{TC}^0$ (only 50% likelihood) than one might expect. (I know of at least one complexity theorist who has worked for years to prove that $\mathsf{NC}^1 = \mathsf{TC}^0$. No, it's not me.) One not-terribly-serious reason for doubting $\mathsf{NC}^1 \neq \mathsf{TC}^0$ is that TC^0 is (as far as we know) the class of circuits most closely resembling the human brain, and we are all familiar with how unexpectedly powerful that sort of computational device can be.

A more serious reason for doubting $\mathsf{NC}^1 \neq \mathsf{TC}^0$ is that NC^1 has proven to be surprisingly easier than expected, and TC^0 has been surprisingly powerful (in a formal sense). Barrington's amazing theorem [Bar89] shows that NC^1 corresponds to only "$O(1)$-space computation" in a computational model that has random access to the input. One corollary is that the word problem over the group S_5 (given a sequence of group elements, is its product the identity?) is already NC^1-complete: solving it in TC^0 would prove $\mathsf{NC}^1 = \mathsf{TC}^0$.

The circuit complexity literature shows that many problems known to be in NC^1 were systematically placed later in TC^0; a nice example is integer division [BCH86,RT92] but many other interesting numerical and algebraic tasks also turn out to be in TC^0 (such as the pseudorandom function constructions of Naor and Reingold [NR04]). For many different types of groups (but not S_5, as far as we know) their word problems are known to be in TC^0 (see [MVW17] for very recent work, with references). In fact, every natural problem that I know in NC^1 is either already NC^1-complete under TC^0 reductions, or is already in TC^0 (there are no good candidate problems which are neither). So perhaps we are one smart threshold circuit away from making the two classes equal.

An argument leaning towards $\mathsf{NC}^1 \neq \mathsf{TC}^0$ may be found in Allender and Koucky [AK10] who show that if $\mathsf{NC}^1 = \mathsf{TC}^0$, then there are $n^{1+\varepsilon}$-size $O(1/\varepsilon)$-depth TC^0 circuits for certain NC^1-complete problems, for every $\varepsilon > 0$. Another point is that, if $\mathsf{NC}^1 = \mathsf{TC}^0$, then (by a padding argument) the class PSPACE lies in the so-called "polynomial-time counting hierarchy" which intuitively seems smaller. Maybe multiple layers of oracle calls to counting solutions are powerful, and such circuits exist? To me, it's a coin flip.

2.10 EXP vs NEXP

Many may wonder why I put only 45% likelihood on $\mathsf{EXP} \neq \mathsf{NEXP}$. (I suspect the others will, instead of wondering, just assume that I'm out of my mind.) Well, for one, we do have that $\mathsf{EXP} \neq \mathsf{NEXP}$ implies $\mathsf{P} \neq \mathsf{NP}$, and the other direction (at least from a provability standpoint) does not seem to hold, so it is natural to consider $\mathsf{EXP} \neq \mathsf{NEXP}$ to be not as likely as $\mathsf{P} \neq \mathsf{NP}$.

To make the percentage dip below 50%, there are other reasons. For one, if we think we're ignorant about what is impossible in P, then we are total idiots about what is impossible in EXP. The scope of what can be done in exponential time has been barely scratched, I believe, and many improved exponential time algorithms in the literature are mainly applications of known polynomial-time strategies to exponential-sized instances. That is, we generally don't know how to construct algorithms that take advantage of the additional structure provided by succinctly-represented inputs — which are the defining feature of many NEXP-complete problems [PY86,GW83]) — and I am inclined to believe that non-trivial algorithms for solving problems on succinct inputs should exist. Arora, Steurer, and Wigderson [ASW09] study exponentially-large graphs whose edge relations are defined by small weak circuits such as AC^0, and give several interesting algorithms for solving problems on such graphs. They also show how the usual NP-complete problems cannot be solved on graphs defined by AC^0 circuits unless $\mathsf{NEXP} = \mathsf{EXP}$.

Let me give a concrete example of the knife edge that NEXP versus EXP sits upon. Let $f : \{0,1\}^{2n} \to \{0,1\}$ be a Boolean function, and define the *graph of* f to be the 2^n-node graph with vertex set $\{0,1\}^n$ and edge set $\{(u,v) \mid f(uv) = 1\}$. Define the $\mathsf{Max\text{-}Clique\text{-}CNF}$ problem to be:

> *Given a CNF F on $2n$ variables and m clauses, and an integer $k \in [2^n]$, is there a clique in the graph of F of size at least k?*

One can define the $\mathsf{Max\text{-}Clique\text{-}DNF}$ problem in an analogous way. In unpublished work, Josh Alman and I noticed that $\mathsf{Max\text{-}Clique\text{-}CNF}$ is solvable in $2^{O(m+n)}$ time, but $\mathsf{Max\text{-}Clique\text{-}DNF}$ is already NEXP-complete, even for constant-width DNFs of n variables and $\mathsf{poly}(n)$ terms! Das, Scharpfenecker, and Torán [DST17] make similar observations for the CNF and DNF versions of other NP-hard problems.

I would be very surprised if the hardest cases of the Max-Clique problem (or even the somewhat-hardest cases) can be generated by tiny DNF formulas of

constant width. I predict that problems solvable in $2^{O(n)}$ nondeterministic time can be solved faster than the naive $2^{2^{O(n)}}$ deterministic running time; perhaps even in $2^{O(n^k)}$ time for some large constant k.

2.11 SETH: The Strong Exponential Time Hypothesis

Recall that SETH asserts that

> For all $\delta < 1$, there is k such that k-SAT on n variables is not in $2^{\delta n}$ time.

So SETH says there is no universal $\delta < 1$ and algorithm solving constant-width CNF SAT instances in $2^{\delta n}$ time. I am generally considered to be a skeptic of SETH (due to work such as [Wil16]), but I am not completely convinced that SETH is false: I put 25% likelihood.

The main reason for skepticism is that the area of exponential-time algorithms has produced *many* results where the naive running time of c^n for an NP-complete problem was reduced to $O((c-\delta)^n)$ for some $\delta > 0$ (see the textbook of Fomin and Kratsch for examples [FK10]). I do not see a good reason for believing that k-SAT is immune to such improvements. But people have tried hard to solve the problem, especially over the last 10 years, so there is some reason to believe SETH. Personally, I have benefited from believing it is false: trying to solve SAT faster has led me down several fruit-bearing paths that I would have never explored otherwise. I believe that contentious conjectures/hypotheses like SETH need to exist, to keep a research area vibrant and active.

I should stress that a large chunk of recent work of the form *SETH implies X* (such as [AVW14,BI15,ABV15,Bri14,BM16,BI16]) does not actually require SETH to be true. In fact, their hardness rests on the following basic *Orthogonal Vectors* (or *Disjoint Sets*) problem.

> Orthogonal Vectors: *Given n Boolean vectors in $c\log(n)$ dimensions (for some constant parameter c), are there two which are orthogonal?*

The *Orthogonal Vectors Conjecture* (OVC) is that for every $\varepsilon > 0$, there is a (potentially large) $c \geq 1$ such that no algorithm solves Orthogonal Vectors in $n^{2-\varepsilon}$ time. It is known that SETH implies OVC [Wil04,WY14], and many SETH-hardness results are actually OVC-hard. It looks very plausible to me that OVC is true but SETH is false.

It is useful to think of the Orthogonal Vectors problem as an interesting detection version of rectangular matrix multiplication: *given a "skinny" Boolean matrix A, does $A \cdot A^T$ contain a zero entry?* Note that detecting if there is a *non-zero* can be done in randomized linear time, by Freivalds' checker for matrix multiplication [Fre77]. So the OVC asks whether matrix multiplication checking in the Boolean domain can be extended to checking for a zero entry, without (essentially) multiplying the two matrices.

2.12 NEXP vs coNEXP

According to Table 1.1, I am putting 80% likelihood on NEXP = coNEXP. Why would a self-respecting complexity theorist do that? Here are a few reasons:

1. **It is true with small advice.** First, it is known that coNEXP is already contained in NEXP *with* $O(n)$ *bits of advice*, as reported by Buhrman, Fortnow, and Santhanam [BFS09]. Given a language $L \in$ coNEXP, for inputs of length n, the advice encodes the number of strings of length n which are in L. Then in nondeterministic exponential time, one can guess the inputs that are *not* in L, guess witnesses for each of them, and verify all of this information. Thus in order to put coNEXP in NEXP without advice, it would suffice to be able to count (in NEXP) the number of accepted strings of length n for an NEXP machine. Note how the power of exponential time is being used: coNP $\subset$ NP/poly seems very unlikely in comparison. This proof looks to be inspired by the inductive counting technique in the proof of NSPACE$[S(n)]$ = coNSPACE$[S(n)]$, due to Immerman [Imm88] and Szelepcsényi [Sze88].
2. **The Spectrum Problem.** A stronger result than NEXP = coNEXP would be implied by an expected resolution of the *spectrum problem*. In finite model theory, the *spectrum* of a first-order sentence ϕ is the set of all finite cardinalities of models of ϕ. For example, if ϕ is a sentence that defines the axioms of a field, then its spectrum is the set of all prime powers. In the 1950s, Asser (see [DJMM12] for a comprehensive survey) asked whether the complement of a spectrum is always a spectrum itself: i.e.,

   > The Spectrum Problem: *Given a first-order sentence ϕ, is there another sentence ψ whose spectrum is the complement of ϕ's spectrum?*

   This question has a long rich history, and some working in finite model theory believe the answer to be yes. Jones and Selman [JS74] showed that the spectrum problem has a yes answer if and only if NTIME$[2^{O(n)}]$ = coNTIME$[2^{O(n)}]$, since in fact the class of all spectra (where the numbers are encoded in some finite alphabet) equals NTIME$[2^{O(n)}]$. There are several interesting conjectures regarding spectra, any of which would imply a yes-answer [Ash94,CM06], and the conclusion would be stronger than proving NEXP = coNEXP. (For instance, one could have nondeterministic $2^{O(n^9)}$-time algorithms for deciding the complements of nondeterministic $O(2^n)$-time problems, and this would still imply NEXP = coNEXP, but not necessarily the spectrum conjecture.)
3. **Max-Clique-DNF.** From the section on EXP vs NEXP (Section 2.10), the following would imply NEXP = coNEXP: Given a DNF formula F of constant width, $2n$ variables, and poly(n) terms, there is a nondeterministic algorithm running in $2^{\text{poly}(n)}$ time which accepts F if and only if the graph of F does not have a clique of a certain desired size. (Recall we said that the corresponding problem for CNF is solvable exactly in $2^{\text{poly}(n)}$ time.)

4. **Why not?** I don't know of any truly counter-intuitive consequences of NEXP = coNEXP. Because one can enumerate over all inputs of length n in exponential time, and in nondeterministic exponential time one can even guess witnesses of exponential length for each input of length n, I think these classes will behave differently than our intuition about lower complexity classes.

2.13 NSETH: Nondeterministic SETH

The NSETH, introduced by Carmosino *et al.* [CGI+16] recently, states:

> For all $\delta < 1$, there is a k such that k-UNSAT on n variables is not in nondeterministic in $2^{\delta n}$ time.

So NSETH proposes that there is no proof system which can refute unsatisfiable $\omega(1)$-width CNFs in $2^{\delta n}$ steps, for any $\delta < 1$.

I put 15% likelihood on NSETH being true. The most obvious reason for skepticism is that the mild extension to Merlin-Arthur proof systems is very false: Formula-UNSAT for $2^{o(n)}$-size formulas can be proved with a probabilistic verifier in only $2^{n/2+o(n)}$ time [Wil16].

Since it is generally believed that MA = NP, one might think the story is essentially over, and that I should have a much lower likelihood for NSETH. That is not quite the case: while MA may well equal NP, it is not clear how a $2^{n/2}$-time MA algorithm could be simulated in 1.999^n nondeterministic time. It's not even clear that the (one-round) Arthur-Merlin version of SETH is false, because the inclusion of MA in AM takes quadratic overhead. Refuting the one-round Arthur-Merlin SETH (where Arthur tosses coins, then Merlin sends a message based on the coins, then an accept/reject decision is made, and we want Merlin to prove that a given formula is UNSAT in 1.999^n time) would probably imply that a non-uniform variant of NSETH is false.

2.14 L vs RL

I put 95% likelihood on L = RL, that is, the problems solvable in randomized logarithmic space equals the problems solvable in (deterministic) logspace. At this moment in time, it feels like this problem has "almost" been solved. Intuitively, there are two factors in favor of L = RL: (1) we already believe that randomness generally does not help solve problems much more efficiently than deterministic algorithms, and (2) space-bounded computation appears to be fairly robust under modifications to the acceptance conditions of the model (think of $\mathsf{NL} \subseteq \mathsf{SPACE}[\log^2 n]$ [Sav70] and NL = coNL [Imm88,Sze88]).

As far as I know, the main problem that was thought to be a potential separator of RL and L was undirected s-t connectivity [AKL+79]. However this

problem was shown to be in L by a remarkable algorithm of Reingold [Rei08]. In follow-up work, Reingold, Trevisan, and Vadhan [RTV06] showed how to solve s-t connectivity in Eulerian directed graphs in L, and show that a logspace algorithm for a seemingly slight generalization of their problem would imply $\mathsf{L} = \mathsf{RL}$. My personal interpretation of these results is that $\mathsf{L} = \mathsf{RL}$ is true, and is only one really good idea away from being resolved.

Acknowledgment. I appreciate Gerhard Woeginger's considerable patience with me during the writing of this article, and Scott Aaronson, Josh Alman, Boaz Barak, Greg Bodwin, Sam Buss, Lance Fortnow, Richard Lipton, Kenneth Regan, Omer Reingold, Gregory Rosenthal, Rahul Santhanam, and Madhu Sudan for helpful comments on a draft, some of which led me to adjust my likelihoods by a few percentage points.

References

Aar16. Scott Aaronson. *P ?= NP. In* Open Problems in Mathematics, pages 1–122. Springer International Publishing, 2016.

ABV15. Amir Abboud, Arturs Backurs, and Virginia Vassilevska Williams. Tight hardness results for LCS and other sequence similarity measures. In *FOCS*, pages 59–78, 2015.

ACW16. Josh Alman, Timothy M. Chan, and R. Ryan Williams. Polynomial representations of threshold functions and algorithmic applications. In *FOCS*, pages 467–476, 2016.

Ajt99. Miklós Ajtai. A non-linear time lower bound for Boolean branching programs. *Theory of Computing*, 1(1):149–176, 2005. Preliminary version in FOCS'99.

AK10. Eric Allender and Michal Koucký. Amplifying lower bounds by means of self-reducibility. *Journal of the ACM*, 57(3), 2010.

AKL$^+$79. Romas Aleliunas, Richard M. Karp, Richard J. Lipton, László Lovász, and Charles Rackoff. Random walks, universal traversal sequences, and the complexity of maze problems. In *FOCS*, pages 218–223, 1979.

ALM$^+$98. Sanjeev Arora, Carsten Lund, Rajeev Motwani, Madhu Sudan, and Mario Szegedy. Proof verification and the hardness of approximation problems. *Journal of the ACM*, 45(3):501–555, 1998.

AS98. Sanjeev Arora and Shmuel Safra. Probabilistic checking of proofs: A new characterization of NP. *Journal of the ACM*, 45(1):70–122, 1998.

Ash94. Christopher J. Ash. A conjecture concerning the spectrum of a sentence. *Math. Log. Q.*, 40:393–397, 1994.

ASW09. Sanjeev Arora, David Steurer, and Avi Wigderson. Towards a study of low-complexity graphs. In *ICALP Part I*, pages 119–131, 2009.

AVW14. Amir Abboud, Virginia Vassilevska Williams, and Oren Weimann. Consequences of faster alignment of sequences. In *ICALP*, pages 39–51, 2014.

AW09. Scott Aaronson and Avi Wigderson. Algebrization: A new barrier in complexity theory. *ACM TOCT*, 1, 2009.

AW17. Josh Alman and Ryan Williams. Probabilistic rank and matrix rigidity. In *STOC*, pages 641–652, 2017.

Bar89. David A. Mix Barrington. Bounded-width polynomial-size branching programs recognize exactly those languages in NC^1. *J. Comput. Syst. Sci.*, 38(1):150–164, 1989.

BCH86. Paul W. Beame, Stephen A. Cook, and H. James Hoover. Log depth circuits for division and related problems. *SIAM Journal on Computing*, 15(4):994–1003, 1986.

BFNW93. László Babai, Lance Fortnow, Noam Nisan, and Avi Wigderson. BPP has subexponential time simulations unless EXPTIME has publishable proofs. *Computational Complexity*, 3(4):307–318, 1993.

BFS09. Harry Buhrman, Lance Fortnow, and Rahul Santhanam. Unconditional lower bounds against advice. In *ICALP Part I*, pages 195–209, 2009.

BGS75. Theodore Baker, John Gill, and Robert Solovay. Relativizations of the P =? NP question. *SIAM Journal on Computing*, 4(4):431–442, 1975.

BI15. Arturs Backurs and Piotr Indyk. Edit distance cannot be computed in strongly subquadratic time (unless SETH is false). In *STOC*, pages 51–58, 2015.

BI16. Arturs Backurs and Piotr Indyk. Which regular expression patterns are hard to match? In *FOCS*, pages 457–466, 2016.

BJS98. Paul Beame, Thathachar S Jayram, and Michael Saks. Time–space tradeoffs for branching programs. *Journal of Computer and System Sciences*, 63(4):542–572, 2001. Preliminary version in FOCS'98.

BM16. Karl Bringmann and Wolfgang Mulzer. Approximability of the discrete Fréchet distance. *JoCG*, 7(2):46–76, 2016.

Bri14. Karl Bringmann. Why walking the dog takes time: Frechet distance has no strongly subquadratic algorithms unless SETH fails. In *FOCS*, pages 661–670, 2014.

BSSV00. Paul Beame, Michael Saks, Xiaodong Sun, and Erik Vee. Time-space tradeoff lower bounds for randomized computation of decision problems. *Journal of the ACM*, 50(2):154–195, 2003. Preliminary version in FOCS'00.

BV02. Paul Beame and Erik Vee. Time-space tradeoffs, multiparty communication complexity, and nearest-neighbor problems. pages 688–697, 2002.

BW12. Samuel R. Buss and Ryan Williams. Limits on alternation trading proofs for time-space lower bounds. *Computational Complexity*, 24(3):533–600, 2015. Preliminary version in CCC'12.

CGI^+16. Marco Carmosino, Jiawei Gao, Russell Impagliazzo, Ivan Mikhailin, Ramamohan Paturi, and Stefan Schneider. Nondeterministic extensions of the Strong Exponential Time Hypothesis and consequences for non-reducibility. In *ACM Conference on Innovations in Theoretical Computer Science (ITCS)*, pages 261–270, 2016.

CM06. Annie Chateau and Malika More. The ultra-weak Ash conjecture and some particular cases. *Math. Log. Q.*, 52(1):4–13, 2006.

COS17. Ruiwen Chen, Igor C. Oliveira, and Rahul Santhanam. An average-case lower bound against ACC^0. *Electronic Colloquium on Computational Complexity (ECCC)*, 24:173, 2017.

Din15. Ning Ding. Some new consequences of the hypothesis that P has fixed polynomial-size circuits. In *TAMC*, pages 75–86. Springer, 2015.

DJMM12. Arnaud Durand, Neil D. Jones, Johann A. Makowsky, and Malika More. Fifty years of the spectrum problem: survey and new results. *Bulletin of Symbolic Logic*, 18(4):505–553, 2012.

DST17. Bireswar Das, Patrick Scharpfenecker, and Jacobo Torán. CNF and DNF succinct graph encodings. *Information and Computation*, 253(3):436–447, 2017.

Dvi17. Zeev Dvir. A generating matrix of a good code may have low rigidity. Written by Oded Goldreich, http://www.wisdom.weizmann.ac.il/~oded/MC/209.pdf, 2017.

FK10. Fedor V Fomin and Dieter Kratsch. *Exact exponential algorithms*. Springer Science & Business Media, 2010.

FLvMV05. Lance Fortnow, Richard J. Lipton, Dieter van Melkebeek, and Anastasios Viglas. Time-space lower bounds for satisfiability. *Journal of the ACM*, 52(6):835–865, 2005.

For00. Lance Fortnow. Time-space tradeoffs for satisfiability. *J. Comput. Syst. Sci.*, 60(2):337–353, 2000.

For15. Lance Fortnow. Nondeterministic separations. In *Theory and Applications of Models of Computation (TAMC)*, pages 10–17, 2015.

Fre77. Rusins Freivalds. Probabilistic machines can use less running time. In *IFIP Congress*, pages 839–842, 1977.

FSW09. Lance Fortnow, Rahul Santhanam, and Ryan Williams. Fixed-polynomial size circuit bounds. In *CCC*, pages 19–26. IEEE, 2009.

Gas02. William I. Gasarch. The P =? NP poll. *ACM SIGACT News*, 33(2):34–47, 2002.

Gas12. William I. Gasarch. Guest column: The second P=? NP poll. *ACM SIGACT News*, 43(2):53–77, 2012.

GM15. Oded Goldreich and Or Meir. Input-oblivious proof systems and a uniform complexity perspective on P/poly. *ACM Transactions on Computation Theory (TOCT)*, 7(4):16, 2015.

GW83. Hana Galperin and Avi Wigderson. Succinct representations of graphs. *Information and Control*, 56(3):183–198, 1983.

Hås01. Johan Håstad. Some optimal inapproximability results. *Journal of the ACM*, 48(4):798–859, 2001.

Hel86. Hans Heller. On relativized exponential and probabilistic complexity classes. *Information and Control*, 71(3):231–243, 1986.

HPV77. John Hopcroft, Wolfgang Paul, and Leslie G. Valiant. On time versus space. *Journal of the ACM*, 24(2):332–337, April 1977.

IKW02. Russell Impagliazzo, Valentine Kabanets, and Avi Wigderson. In search of an easy witness: Exponential time vs. probabilistic polynomial time. *J. Comput. Syst. Sci.*, 65(4):672–694, 2002.

Imm88. Neil Immerman. Nondeterministic space is closed under complement. *SIAM Journal on Computing*, 17:935–938, 1988.

IW97. Russell Impagliazzo and Avi Wigderson. P = BPP if E requires exponential circuits: Derandomizing the XOR lemma. In *STOC*, pages 220–229, 1997.

JS74. Neil D. Jones and Alan L. Selman. Turing machines and the spectra of first-order formulas. *J. Symb. Log.*, 39(1):139–150, 1974.

Kan82. Ravi Kannan. Circuit-size lower bounds and non-reducibility to sparse sets. *Information and Control*, 55(1):40–56, 1982.

Kho02. Subhash Khot. On the power of unique 2-prover 1-round games. In *STOC*, pages 767–775. ACM, 2002.

KMS17. Subhash Khot, Dor Minzer, and Muli Safra. On independent sets, 2-to-2 games, and grassmann graphs. In *STOC*, pages 576–589, 2017.

KMS18. Subhash Khot, Dor Minzer, and Muli Safra. Pseudorandom sets in grassmann graph have near-perfect expansion. *Electronic Colloquium on Computational Complexity (ECCC)*, 18(6), 2018.

KW98. Johannes Kobler and Osamu Watanabe. New collapse consequences of NP having small circuits. *SIAM Journal on Computing*, 28(1):311–324, 1998.

Lip94. Richard J. Lipton. Some consequences of our failure to prove non-linear lower bounds on explicit functions. In *Structure in Complexity Theory Conference*, pages 79–87, 1994.

Lip10. Richard J. Lipton. *The P=NP Question and Gödel's Lost Letter*. Springer Science & Business Media, see also http://rjlipton.wordpress.com, 2010.

LR13. Richard J. Lipton and Kenneth W. Regan. *People, Problems, and Proofs*. Springer, see also http://rjlipton.wordpress.com, 2013.

LW13. Richard J. Lipton and Ryan Williams. Amplifying circuit lower bounds against polynomial time, with applications. *Computational Complexity*, 22(2):311–343, 2013.

MVW17. Alexei Miasnikov, Svetla Vassileva, and Armin Weiß. The conjugacy problem in free solvable groups and wreath products of abelian groups is in TC0. In Pascal Weil, editor, *International Computer Science Symposium in Russia (CSR)*, pages 217–231. Springer International Publishing, 2017.

NR04. Moni Naor and Omer Reingold. Number-theoretic constructions of efficient pseudo-random functions. *Journal of the ACM*, 51(2):231–262, 2004.

NW94. Noam Nisan and Avi Wigderson. Hardness vs randomness. *J. Comput. Syst. Sci.*, 49(2):149–167, 1994.

PY86. Christos H. Papadimitriou and Mihalis Yannakakis. A note on succinct representations of graphs. *Information and Control*, 71(3):181–185, 1986.

Rei08. Omer Reingold. Undirected connectivity in log-space. *Journal of the ACM*, 55(4):17:1–17:24, September 2008.

RR97. Alexander Razborov and Steven Rudich. Natural proofs. *J. Comput. Syst. Sci.*, 55(1):24–35, 1997.

RT92. John H. Reif and Stephen R. Tate. On threshold circuits and polynomial computation. *SIAM Journal on Computing*, 21(5):896–908, 1992.

RTV06. Omer Reingold, Luca Trevisan, and Salil Vadhan. Pseudorandom walks on regular digraphs and the RL vs L problem. In *STOC*, pages 457–466. ACM, 2006.

San07. Rahul Santhanam. Circuit lower bounds for Merlin–Arthur classes. volume 39, pages 1038–1061, 2009. Preliminary version in STOC'07.

Sav70. Walter J. Savitch. Relationships between nondeterministic and deterministic tape complexities. *J. Comput. Syst. Sci.*, 4(2):177–192, April 1970.

Sze88. Róbert Szelepcsényi. The method of forced enumeration for nondeterministic automata. *Acta Informatica*, 26(3):279–284, November 1988.

Tam16. Suguru Tamaki. A satisfiability algorithm for depth two circuits with a sub-quadratic number of symmetric and threshold gates. *Electronic Colloquium on Computational Complexity (ECCC)*, 23:100, 2016.

Wil08a. R. Ryan Williams. Time-space tradeoffs for counting NP solutions modulo integers. *Computational Complexity*, 17(2):179–219, 2008.

Wil08b. Ryan Williams. Non-linear time lower bound for (succinct) quantified boolean formulas. *Electronic Colloquium on Computational Complexity (ECCC)*, (TR08-076), 2008.

Wil13a. Ryan Williams. Alternation-trading proofs, linear programming, and lower bounds. *TOCT*, 5(2):6, 2013.

Wil13b. Ryan Williams. Towards NEXP versus BPP? In *International Computer Science Symposium in Russia (CSR)*, pages 174–182. Springer, 2013.

Wil14. Ryan Williams. New algorithms and lower bounds for circuits with linear threshold gates. In *STOC*, pages 194–202, 2014.

Wil16. Richard Ryan Williams. Strong ETH breaks with Merlin and Arthur: Short non-interactive proofs of batch evaluation. In *CCC*, pages 2:1–2:17, 2016.

Wil11. Ryan Williams. Nonuniform ACC circuit lower bounds. *JACM*, 61(1):2, 2014. Preliminary version in CCC'11.

Wil04. Ryan Williams. A new algorithm for optimal 2-constraint satisfaction and its implications. *Theor. Comput. Sci.*, 348(2-3):357–365, 2005. Preliminary version in ICALP'04.

Wil10. Ryan Williams. Improving exhaustive search implies superpolynomial lower bounds. *SIAM Journal on Computing*, 42(3):1218–1244, 2013. Preliminary version in STOC'10.

WY14. Ryan Williams and Huacheng Yu. Finding orthogonal vectors in discrete structures. In *SODA*, pages 1867–1877, 2014.