The Logic in Computer Science Column

by

Yuri Gurevich

Microsoft Research
One Microsoft Way, Redmond WA 98052, USA
gurevich@microsoft.com

Truth vs. Proof in Computational Complexity

Boaz Barak[*]

Theoretical Computer Science is blessed (or cursed?) with many open problems. For some of these questions, such as the P vs NP problem, it seems like it could be decades or more before they reach resolution. So, if we have no proof either way, what do we assume about the answer? We could remain agnostic, saying that we simply don't know, but there can be such a thing as *too much* skepticism in science. For example, Scott Aaronson once claimed [2] that in other sciences $P \neq NP$ would by now have been declared a law of nature. I tend to agree. After all, we are trying to uncover the truth about the nature of computation and this quest won't go any faster if we insist on discarding all evidence that is not in the form of mathematical proofs from first principles.

But what other methods can we use to get evidence for questions in computational complexity? After all, it seems completely hopeless to experimentally verify even a non-asymptotic statement such as "There is no circuit of size 2^{100} that can solve

[*] Microsoft Research New England, Cambridge, MA, b@boazbarak.org. Adapted from a post at the "Windows on Theory" blog http://windowsontheory.org/

Bulletin of the EATCS no 108, pp. 130–142, October 2012
© European Association for Theoretical Computer Science

3SAT on 10,000 variables". There is in some sense only one tool us scientists can use to predict the answer to open questions, and this is Occam's Razor. That is, if we want to decide whether an answer to a certain question is *Yes* or *No*, we try to think of the simplest/nicest possible world consistent with our knowledge in which the answer is Yes, and the simplest such world in which the answer is No. If one of these worlds is much nicer than the other, that would suggest that it is probably the true one. For example, if assuming the answer to the question is "Yes" yields several implications that have been independently verified, while we must significantly contort the "No" world in order to make it consistent with current observations, then it is reasonable to predict that the answer is "Yes".

In this essay, I attempt to do this exercise for two fascinating conjectures for which, unlike the P vs NP problem, there is no consensus on their veracity: Khot's *Unique Games Conjecture* [23] and Feige's *Random 3SAT Hypothesis* [16]. This is both to illuminate the state of the art on these particular conjectures, and to discuss the general issue of what can be considered as valid evidence for open questions in computational complexity.

1 The Unique Games Conjecture

Khot's *Unique Games Conjecture* (UGC) [23] states that a certain approximation problem (known as "Unique Games" or UG) is NP hard. I'll define the UG problem below, but one benefit of using Occam's Razor is that we can allow ourselves to discuss a closely related problem known as *Small Set Expansion* (SSE), which I find more natural than the UG problem. The SSE problem can be described as the problem of "finding a cult inside a social network":[1] you're given a graph G over n vertices, and you know that it contains a set S of at most, say, $n/\log n$ vertices that is "almost isolated" from the rest of the graph, in the sense that a typical member of S has 99% of its neighbors also inside S. The goal is to find S or any set S' of similar size that is reasonably isolated (say having more than half of the neighbors inside it). Formally, for every $\epsilon, \delta > 0$ and number k, the computational problem $\mathsf{SSE}(\epsilon, \delta, k)$ is to distinguish, given a d-regular graph $G = (V, E)$, between the case that there is a set $S \subseteq V$ with $|S| \leq |V|/k$ and with $|E(S, S)| \geq (1 - \epsilon)\delta d|S|$, and the case that for every $S \subseteq V$ with $|S| \leq |V|/k$, $|E(S, S)| \leq \delta|S|$. The following conjecture seems very closely related to the unique games conjecture:

[1] I use the term "cult" since we're looking for a set in which almost all connections stay inside. In contrast, a "community" would correspond to a set containing a higher than expected number of connections. The computational problem associated with finding such a "community" is the *densest k-subgraph* problem [17, 9, 10], and it seems considerably harder than either the UG or SSE problems.

Conjecture 1 (Small Set Expansion Hypothesis (SSEH) [33])**.** *For every $\epsilon, \delta > 0$ there exists k such that* $\mathsf{SSE}(\epsilon, \delta, k)$ *is NP-hard.*

Almost all that I'll say in this essay will hold equally well for the SSE and UG problems, and so the reader can pretend that the Unique Games Conjecture is the same as the Small Set Expansion Hypothesis without much loss in understanding. But for the sake of accuracy and completeness, I'll now define the Unique Games problem, and explain some of its relations to the SSE problem. The UG problem is also parameterized by ϵ, δ, k. The input for the $\mathsf{UG}(\epsilon, \delta, k)$ problem is a set of m equations on n variables $x_1, \ldots, x_n$ over the alphabet $[k] = \{1, \ldots, k\}$. Each equation has the form $x_i = \pi_{i,j}(x_j)$, where $\pi_{i,j}$ is a permutation over $[k]$. The computational task is to distinguish between the case that there is an assignment to the variables that satisfies at least $(1 - \epsilon)m$ equations, and the case that no assignment satisfies more than δm of them. The formal statement of the *Unique Games Conjecture* is the following:

Conjecture 2 (Unique Games Conjecture (UGC) [23])**.** *For every $\epsilon, \delta > 0$ there exists k such that* $\mathsf{UG}(\epsilon, \delta, k)$ *is NP-hard.*

One relation between the UG and SSE problems is that we can always transform an instance Ψ of UG into the graph G_Ψ on the vertex set $V = [n] \times [k]$ containing an edge between the vertices (i, a) and (j, b) (for $i, j \in [n]$ and $a, b \in [k]$) if and only if there is an equation in Ψ of the form $a_i = \pi_{i,j}(x_j)$ with $a = \pi_{i,j}(b)$. Now, an assignment $\sigma \in [k]^n$ to the variables of Ψ will translate naturally into a set $S_\sigma \subseteq V(G_\Psi)$ of $n = |V|/k$ vertices containing the vertex (i, a) iff $\sigma_i = a$. It can be easily verified that edges with both endpoints in S_σ will correspond exactly to the equations of Ψ that are satisfied by σ. One can show that the UG problem has the same difficulty if every variable in Ψ participates in the same number d of equations, and hence the map $\Psi \mapsto G_\Psi$ transforms a $\mathsf{UG}(\epsilon, \delta, k)$ instance into an $\mathsf{SSE}(\epsilon, \delta, k)$ instance, and in fact maps the "Yes case" of $\mathsf{UG}(\epsilon, \delta, k)$ into the "Yes case" of $\mathsf{UG}(\epsilon, \delta, k)$. Alas, this is not a reduction from UG to SSE, because it can map a "No" instance of UG into a "Yes" instance of SSE. In fact, the only reduction known between the problems is in the other direction: Raghavendra and Steurer [33] showed that SSE is no harder than UG and hence the UGC implies the SSEH. However, all the known algorithmic and hardness results hold equally well for SSE and UG [32, 34, 3, 14, 35, 8], strongly suggesting that these problems have the same computational difficulty. Hence in this essay I will treat them as equivalent.

Let us now turn to exploring how natural is the world where the UGC (or SSEH) holds, versus the world in which it fails.

1.1 The "UGC true" world.

There is one aspect in which the world where the UGC is true is very nice indeed. One of the fascinating phenomenons of complexity is the *dichotomy* exhibited by many natural problems: they either have a polynomial-time algorithm (often with a low exponent) or are NP-hard, with very few examples in between. A striking result of Raghavendra [31] showed that the UGC implies a beautiful dichotomy for a large family of problems, namely the constraint-satisfaction problems (CSP). He showed, that for every CSP P, there is a number $\alpha_{\mathsf{UG}}(P)$ (which we'll call the UG *threshold* of P), such that for every $\epsilon > 0$, the problem of maximizing the satisfied constraints of an instance of P can be approximated within $\alpha_{UG}(P) - \epsilon$ in polynomial (in fact, quasilinear [37]) time, while if the UGC is true, then achieving an $\alpha_{\mathsf{UG}}(P) + \epsilon$ approximation is NP hard.

This is truly a beautiful result, but alas there is one wrinkle in this picture: where you might expect that in this dichotomy the hard problems would all be equally hard, there is a subexponential algorithm for unique games [3] showing that if the UGC is true then some constraint satisfaction problems can be solved in time 2^{n^ϵ} for some $\epsilon \in (0, 1)$. While those sub-exponential problems are asymptotically hard, compared to "proper" NP-hard problems such as SAT, the input sizes when the asymptotic hardness 'kicks in" will be pretty huge. For example, for the Small Set Expansion problem with the parameters above (99% vs 50% approximation), the [3] algorithm will take roughly $2^{n^{1/10}}$ steps which is pretty efficient for graphs with up to 2^{60} or so vertices.

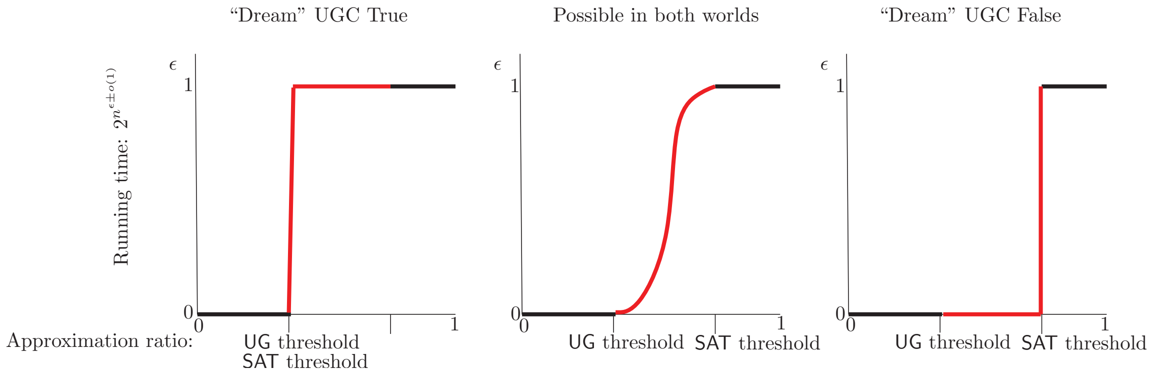
In more qualitative terms, many hardness of approximation results for CSP's actually use *quasilinear* reductions from SAT [30], and so let us define the SAT *threshold* of P, to be the the smallest number $\alpha_{\mathsf{SAT}}(P)$ for which achieving an $\alpha_{\mathsf{SAT}}(P) + \epsilon$ approximation is NP-hard via a quasilinear reduction. In a "dream version" of the UGC, we may have expected that the NP hardness for UG would use a quasilinear reduction as well, in which case (since Raghavendra's result uses a quasilinear reduction) it would show that $\alpha_{\mathsf{UG}}(P) = \alpha_{\mathsf{SAT}}(P)$ for all P. In particular, assuming the Exponential Time Hypothesis [22] (namely, the assumption that SAT can't be solved in $2^{o(n)}$ time), that would have implied that getting a better than $\alpha_{\mathsf{UG}}(P)$ approximation for P takes $2^{n^{1-o(1)}}$ time— essentially as much time as taken by the brute force algorithm. However, the subexponential time algorithm for UG rules out the possibility of such a reduction, and shows that if the UGC is true, then at least for some CSP's the SAT threshold will be strictly larger than the UG threshold, and the time to approximate them will grow much more gradually with the approximation quality, see Figure 1. Whether such a gradual time/quality tradeoff is more or less beautiful than a sharp jump is in the eyes of the beholder, but it does show that the "dichotomy" picture is more complex than what it initially appears to be.

Raghavendra's theorem is perhaps one reason to wish that the UGC was true, but how does the UGC mesh with current knowledge? One obvious way in which the current state of the art supports the conjecture is that we don't know of any algorithm that refutes it by solving the SSE or UG problems (or any other problem they have been reduced to). However, by this we mean that there is no algorithm *proven* to solve the problem on *all* instances. So there has been an ongoing "battle" between papers showing algorithms that work for natural instances, and papers showing instances that fool natural algorithms. For example, the basic semidefinite program (which is a natural analog of the Geomans-Williamson semidefinite program for Max-Cut [20]) solves the problem on random or expanding input graphs [6]. On the other hand, it was shown that there are instances fooling this program [29] (and some generalizations [32, 28, 27]), along the way disproving a conjecture of Goemans and Linial. The subexponential algorithm mentioned above actually runs much faster on those instances [26], and so for some time I thought it might actually be a *quasi-polynomial time* algorithm. But it turned out there are instances (based on the Reed-Muller code) that require it to take (almost) subexponential time [11]. Nevertheless, the latest round in this battle was won by the algorithmic side: it turned out that all those papers showing hard instances utilized arguments that can be captured by a *sum of squares* formal proof system, which implies that the stronger "Sum-of-Squares"/"Lasserre" semidefinite programming hierarchy[2] can solve them in polynomial time [8]. The latest result also showed connections between the SSE problem and the problems of optimizing hypercontractive norms of operators (norms of the form $p \to q$ for $q > p$) and the *injective tensor norm* problem that arises in quantum information theory.

1.2 The "UGC False" world.

While the Unique Games Conjecture can fail in a myriad of ways, the simplest world in which it fails is that there is an efficient (say polynomial or quasipolynomial time) algorithm for the SSE and UG problems. And, given current knowledge, the natural candidate for such an algorithm comes from the "Sum-of-Squares" semidefinite programming hierarchy mentioned above. Indeed, given that SSE is such a fairly natural problem on graphs, it's quite mind boggling that finding hard instances for it has been so difficult. Contrast this with seemingly related prob-

[2] A semidefinite programming hierarchy is obtained by systematically strengthening a basic semidefinite program with additional constraints. Such hierarchies are parameterized by a number r of rounds, and optimizing over the r^{th} rounds of the hierarchy takes $n^{O(r)}$ time; see [15] for a recent survey.



For every CSP P, we can achieve an approximation ratio $\alpha < \alpha_{\mathsf{UG}}(P)$ in $2^{n^{o(1)}}$ (in fact $\tilde{O}(n)$) time, and it is believed (assuming the Exponential Time Hypothesis) that achieving approximation ratio of $\alpha > \alpha_{\mathsf{SAT}}(P)$ takes $2^{n^{1-o(1)}}$ time, where $\alpha_{\mathsf{UG}}(P)$ and $\alpha_{\mathsf{SAT}}(P)$ are the UG and SAT thresholds defined in Section 1.1. Generally speaking, we do not know the running time required for achieving approximation ratios in the interval $(\alpha_{\mathsf{UG}}(P), \alpha_{\mathsf{SAT}}(P))$ and it can range between the three scenarios depicted above. However, the subexponential time algorithm for UG rules out the "dream UGC true" picture for at least *some* CSP's. Note that sometimes the UG threshold and SAT threshold coincide (e.g., for 3SAT they both equal 7/8). For such CSP's, regardless of whether the UGC is true, it is believed that the time/ratio curve has a discontinuous jump from running time $2^{n^{o(1)}}$ to time $2^{n^{1-o(1)}}$.

Figure 1: Possible running time exponents vs. approximation quality curves for a constraint satisfaction problem.

lems such as densest k-Subgraph, where random instances seem to be the hardest case. On the other hand, we already know that random instances are *not* the hardest ones for SSE, so perhaps those hard instances do exist somewhere and will eventually be found.

There is actually a spectrum of problems "in the unique games flavor" including not just SSE and UG, but also *Max-Cut*, *Balanced Separator* and many others. The cleanest version of the "UGC False" world would be that all of these are significantly easier than NP-hard problems, but whether they are all equal in difficulty is still unclear. In particular, while in the "UGC False" world there will be a $2^{n^{o(1)}}$-time approximation for some CSP's beyond the UG threshold, even the qualitative behavior of the running time/approximation ratio curve is not known (i.e., does it look like the middle or rightmost scenario in Figure 1?).

1.3 A personal bottom line.

Regardless of whether the UGC is true, it has been an extremely successful conjecture in that it led to the development of many new ideas and techniques that have found other applications. I am certain that it will lead to more such ideas before it is resolved. There are a number of ways that we could get more confidence in one of the possibilities. Interestingly, both in the "UGC True" and "UGC False" worlds, our current best candidate for the algorithm meeting the time/ratio curve is the "Sum of Squares" semidefinite programming hierarchy. So, in my mind, finding out the approximation quality for SSE of, say, polylog(n) rounds (corresponding to $n^{\mathrm{polylog}(n)}$ running time) of this hierarchy is a pivotal question. Finding instances that fool this algorithm would go a long way toward boosting confidence in the "UGC True" case, especially given that doing so would require using new ideas beyond sum-of-squares arguments. Another way to support the UGC is to try to come up with candidate NP-hardness reductions (even without analysis or assuming some gadgets that have yet to be constructed) for proving it, or to show NP-hardness for problems such as Max-Cut that are "morally close" to the UG/SSE questions. On this latter point, there are some hardness results for problems such as 3LIN over the reals [25], L_p subspace approximation [19], and subspace hypercontractivity [8] that have some relation to the UG/ SSE, but whether they can be thought having "morally equivalent" complexity to UG/ SSE is still very much in question. To get confidence in the "UGC False" case we can try to show that a smallish number of rounds of the sum-of-squares hierarchy can solve the SSE on a larger family of instances than what is currently known. A pet question of mine is to show that this algorithm works on all Cayley graphs over the Boolean cube. I think that showing this would require ideas that may enable solving the general case as well. Indeed, my current guess is that the UGC is false and that the sum-of-squares algorithm does solve the problem in a reasonable (e.g., quasipolynomial) time.

2 Feige's Random 3SAT Hypothesis

Unlike Khot's conjecture, *Feige's Hypothesis* (FH) [16] deals with *average-case complexity*. While a counting argument easily shows that with high probability a random 3SAT formula on n variables and $1000n$ clauses will not be (even close to) satisfiable, the hypothesis states that there is no efficient algorithm that can *certify* this fact. Formally, the conjecture is defined as follows:

Conjecture 3 (Feige's Hypothesis, weak version[3] [16])**.** *For every $\epsilon > 0$, $d \in \mathbb{N}$, and polynomial-time algorithm A that can output either "$\mathtt{SAT}$" or "$\mathtt{UNSAT}$", it holds that for sufficiently large n, either*

- $\Pr[A(\varphi) = \mathtt{UNSAT}] < 1/2$*, where φ is a random 3SAT formula with n variables and dn clauses.*

or

- *There exists a formula φ on n variables such that there is an assignment satisfying $\geq 1 - \epsilon$ fraction of φ's clauses, but $A(\varphi) = \texttt{UNSAT}$.*

That is, any one-sided error algorithm for 3SAT (i.e., an algorithm that can sometimes say SAT on an unsatisfiable instance, but will never say UNSAT on a nearly satisfiable one) will (wrongly) answer SAT on a large fraction of the input formulas. Feige's hypothesis (and variants of similar flavor [7, 1]) have been used to derive various hardness of approximation results. Applebaum, Wigderson and I [4] also used related (though not equivalent) assumptions to construct a public-key cryptosystem, with the hope that basing cryptosystems on such combinatorial problems will make them immune to algebraic and/or quantum attaches. We note that while the conjecture was originally stated for 3SAT, in a recent manuscript with Kindler and Steurer [12] we show that it can be generalized to every constraint satisfaction problem. Personally I find the k-XOR predicate (i.e., noisy sparse linear equations) to be the cleanest version.

There is added motivation for trying to study heuristic evidence (as opposed to formal proofs) for Feige's hypothesis. Unlike the UGC, which in principle can be proven via a PCP-type NP-hardness reduction of the type we've seen before, proving FH seems way beyond our current techniques (even if we're willing to assume standard assumptions such $P \neq NP$, the existence of one-way functions, or even the hardness of integer factoring). Thus if Feige's hypothesis is true, our only reasonable hope is to show that this holds is by a physics-like process of accumulating evidence, rather than by a mathematical proof. Let us now try to examine this evidence:

2.1 The "FH True" world.

One natural way to refute Feige's Hypothesis would be to show a 0.88 (worst-case) approximation algorithm for 3SAT. This is an algorithm B that given a formula for which an α fraction of the clauses can be satisfied, returns an assignment satisfying 0.88α of them. In particular, given as input a *satisfiable* formula, B must return an assignment satisfying at least 0.88 fraction of the clauses. Thus, we can transform B into a one-sided error algorithm A that answers SAT on an instance if and only if B returns such a 0.88-satisfying assignment for it. Since in a random 3SAT formula, the maximum fraction of satisfiable clauses is very close to $7/8 = 0.875$,

[3] I call this the *weak* version since Feige also phrased a version of the hypothesis with $\epsilon = 0$. However, I prefer the $\epsilon > 0$ version as it is more robust and can be applied to other predicates such as XOR.

the algorithm A would refute FH. However, Håstad's seminal result [21] shows that 3SAT doesn't have such a 0.88-approximation algorithm, hence giving at least some evidence for the "FH True" world.

Feige showed that his hypothesis implies several other such hardness of approximation results, including some not known before to hold under $P \neq NP$; deriving such results was Feige's motivation for the hypothesis. But the connection also works in the other direction: verifying the hardness-of-approximation predictions of FH can be viewed as giving evidence to the "FH True" world, particulary when (as was the case in [24]) the hardness of approximation results were obtained *after* Feige's predictions.

Of course, these hardness of approximation results only relate to *worst-case* complexity while the average-case problem could be potentially much easier. We do note however that in many of these cases, these hardness results are believed to hold even with respect to *subexponential* (e.g. $2^{o(n)}$ or perhaps $2^{n^{1-\Omega(1)}}$) time algorithms. While this doesn't imply average-case hardness, it does mean that the set of hard instances cannot be *too* small. Moreover, the most natural candidate algorithms to refute Feige's hypothesis— the same sum-of-squares relaxations mentioned above— are known [18, 36] not to succeed in certifying unsatisfiability of random instances. Also, as Feige shows, this problem is related to random noisy 3XOR equations, which is a sparse version of the known and well studied Learning Parity with Noise problem (see also discussion in [7, 4]).

The world in which the generalized form [12] of FH holds is particularly nice in that there is a single algorithm (in fact, the same Goemans-Williamson semidefinite program from above) that achieves the optimal performance on every random constraint-satisfaction problem.

2.2 The "FH False" world.

If Feige's Hypothesis is false, then there should be an algorithm refuting it. No such algorithm is currently known. This could be viewed as significant evidence for FH, but the question is how hard people have tried. Random 3-SAT instances (and more generally k-SAT or other CSP's) are actually widely studied and are of interest to physicists, and (with few hundred variables) are also part of SAT solving competitions. But the instances studied are typically in the *satisfiable* regime where the number of clauses is sufficiently small (e.g., less than $\sim 4.26n$ for 3-SAT) so solutions will actually exist. The survey propagation algorithm [13] does seem to work very well for satisfiable random 3SAT instances, but it does not seem to be applicable in the unsatisfiable range. Survey propagation also seems to fail on other CSPs, including k-SAT for $k > 3$ [5].

While not known to be equivalent, there is a variant of FH where the goal is not to certify unsatisfiability of a random 3SAT but to find a planted nearly satisfying assignment of a random 3XOR instance. Such instances might be more suitable for computational challenges (a la the RSA Factoring challenge) as well as SAT solving competitions. It would be interesting to study how known heuristics fare on such inputs.

2.3 A personal bottom line.

Unlike worst-case complexity, our understanding of average-case complexity is very rudimentary. This has less to do with the importance of average-case complexity, which is deeply relevant not just for studying heuristics but also for cryptography, statistical physics, and other areas, and more to do with the lack of mathematical tools to handle it. In particular, almost every hardness reduction we know of uses *gadgets* which end up skewing the distribution of instances. I believe studying Feige's Hypothesis and its ilk (including the conjectures that solution-space shattering implies hardness [5]) offer some of our best hopes for more insight into average-case complexity. I don't know if we will be able to establish a similar web of reductions to the one we have for worst-case complexity, but perhaps we can come up with meta-conjectures or principles that enable us to predict where the line between easiness and hardness will be drawn in each case. We can study the truth of such conjectures using a number of tools, including not just algorithms and reductions but also integrality-gap proofs, physics-style analysis of algorithms, worst-case hardness-of-approximation results, and actual computational experiments.

As for the truth of Feige's Hypothesis itself, while it would be premature to use an FH-based encryption to protect state secrets, I think the current (admittedly inconclusive) evidence points in the direction of the hypothesis being true. It definitely seems as if refuting FH would require a new and exciting algorithmic idea. With time, if Feige's Hypothesis receives the attention it deserves then we can get more confidence in its veracity, or learn more about algorithms for average-case instances.

Parting thoughts

Theoretical Computer Science is sometimes criticized for its reliance on unproven assumptions, but I think we'll need many more of those if we want to get further insights into areas such as average-case complexity. Sure, this means we have to live with possibility that our assumptions turn out to be false, just as physicists have to live with the possibility that future experiments might require a revision of the laws of nature. But that doesn't mean that we should let unconstructive skepticism paralyze us. It would be good if our field had more explicit discussion of what kinds of results can serve as evidence for the hardness or easiness of a computational problem. I deliberately chose two questions whose answer is yet unclear, and for which there is reasonable hope that we'll learn new insights in the coming years that may upend current beliefs. I hope that as such results come to light, we can reach a better understanding of how we can predict the answer to questions for which we have yet no proofs.

References

[1] Noga Alon, Sanjeev Arora, Rajsekar Manokaran, Dana Moshkovitz, and Omri Weinstein. Inapproximability of densest k-subgraph from average case hardness. Manuscript, available at `www.cs.princeton.edu/~rajsekar/papers/dks.pdf`, 2011.

[2] Scott Aaronson. Has there been progress on the P vs. NP question?, 2010. Presentation at MIT CSAIL student workshop. Slides available at `http://www.scottaaronson.com/talks/pvsnpcsw.ppt`.

[3] Sanjeev Arora, Boaz Barak, and David Steurer. Subexponential algorithms for unique games and related problems. In *FOCS*, pages 563–572, 2010.

[4] Benny Applebaum, Boaz Barak, and Avi Wigderson. Public-key cryptography from different assumptions. In *STOC*, pages 171–180, 2010.

[5] Dimitris Achlioptas and Amin Coja-Oghlan. Algorithmic barriers from phase transitions. In *FOCS*, pages 793–802, 2008.

[6] Sanjeev Arora, Subhash Khot, Alexandra Kolla, David Steurer, Madhur Tulsiani, and Nisheeth K. Vishnoi. Unique games on expanding constraint graphs are easy: extended abstract. In *STOC*, pages 21–28, 2008.

[7] Michael Alekhnovich. More on average case vs approximation complexity. *Computational Complexity*, 20(4):755–786, 2011. Preliminary version in FOCS 2003.

[8] Boaz Barak, Fernando G. S. L. Brandão, Aram Wettroth Harrow, Jonathan A. Kelner, David Steurer, and Yuan Zhou. Hypercontractivity, sum-of-squares proofs, and their applications. In *STOC*, pages 307–326, 2012.

[9] Aditya Bhaskara, Moses Charikar, Eden Chlamtac, Uriel Feige, and Aravindan Vijayaraghavan. Detecting high log-densities: an $o(n^{1/4})$ approximation for densest k-subgraph. In *STOC*, pages 201–210, 2010.

[10] Aditya Bhaskara, Moses Charikar, Aravindan Vijayaraghavan, Venkatesan Guruswami, and Yuan Zhou. Polynomial integrality gaps for strong sdp relaxations of densest k-subgraph. In *SODA*, pages 388–405, 2012.

[11] Boaz Barak, Parikshit Gopalan, Johan Håstad, Raghu Meka, Prasad Raghavendra, and David Steurer. Making the long code shorter. In *FOCS*, 2012.

[12] Boaz Barak, Guy Kindler, and David Steurer. On the optimality of relaxations for average-case and generalized constraint satisfaction problems. Manuscript, available from `http://www.boazbarak.org/research.html`, 2012.

[13] Alfredo Braunstein, Marc Mézard, and Riccardo Zecchina. Survey propagation: An algorithm for satisfiability. *Random Struct. Algorithms*, 27(2):201–226, 2005.

[14] Boaz Barak, Prasad Raghavendra, and David Steurer. Rounding semidefinite programming hierarchies via global correlation. In *FOCS*, pages 472–481, 2011.

[15] Eden Chlamtac and Madhur Tulsiani. Convex relaxations and integrality gaps, 2010. Chapter in Handbook on Semidefinite, Cone and Polynomial Optimization.

[16] Uriel Feige. Relations between average case complexity and approximation complexity. In *STOC*, pages 534–543, 2002.

[17] Uriel Feige, David Peleg, and Guy Kortsarz. The dense k-subgraph problem. *Algorithmica*, 29(3):410–421, 2001.

[18] Dima Grigoriev. Linear lower bound on degrees of positivstellensatz calculus proofs for the parity. *Theor. Comput. Sci.*, 259(1-2):613–622, 2001.

[19] Venkatesan Guruswami, Prasad Raghavendra, Rishi Saket, and Yi Wu. Bypassing ugc from some optimal geometric inapproximability results. In *SODA*, pages 699–717, 2012.

[20] Michel X. Goemans and David P. Williamson. Improved approximation algorithms for maximum cut and satisfiability problems using semidefinite programming. *J. ACM*, 42(6):1115–1145, 1995.

[21] Johan Håstad. Some optimal inapproximability results. *J. ACM*, 48(4):798–859, 2001.

[22] Russell Impagliazzo, Ramamohan Paturi, and Francis Zane. Which problems have strongly exponential complexity? *J. Comput. Syst. Sci.*, 63(4):512–530, 2001.

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

[24] Subhash Khot. Ruling out ptas for graph min-bisection, densest subgraph and bipartite clique. In *FOCS*, pages 136–145, 2004.

[25] Subhash Khot and Dana Moshkovitz. Np-hardness of approximately solving linear equations over reals. In *STOC*, pages 413–420, 2011.

[26] Alexandra Kolla. Spectral algorithms for unique games. In *IEEE Conference on Computational Complexity*, pages 122–130, 2010.

[27] Subhash Khot, Preyas Popat, and Rishi Saket. Approximate lasserre integrality gap for unique games. In *APPROX-RANDOM*, pages 298–311, 2010.

[28] Subhash Khot and Rishi Saket. Sdp integrality gaps with local ℓ_1-embeddability. In *FOCS*, pages 565–574, 2009.

[29] Subhash Khot and Nisheeth K. Vishnoi. The unique games conjecture, integrality gap for cut problems and embeddability of negative type metrics into l_1. In *FOCS*, pages 53–62, 2005.

[30] Dana Moshkovitz and Ran Raz. Two-query pcp with subconstant error. *J. ACM*, 57(5), 2010.

[31] Prasad Raghavendra. Optimal algorithms and inapproximability results for every csp? In *STOC*, pages 245–254, 2008.

[32] Prasad Raghavendra and David Steurer. Integrality gaps for strong sdp relaxations of unique games. In *FOCS*, pages 575–585, 2009.

[33] Prasad Raghavendra and David Steurer. Graph expansion and the unique games conjecture. In *STOC*, pages 755–764, 2010.

[34] Prasad Raghavendra, David Steurer, and Prasad Tetali. Approximations for the isoperimetric and spectral profile of graphs and related parameters. In *STOC*, pages 631–640, 2010.

[35] Prasad Raghavendra, David Steurer, and Madhur Tulsiani. Reductions between expansion problems. In *IEEE Conference on Computational Complexity*, pages 64–73, 2012.

[36] Grant Schoenebeck. Linear level lasserre lower bounds for certain k-csps. In *FOCS*, pages 593–602, 2008.

[37] David Steurer. Fast sdp algorithms for constraint satisfaction problems. In *SODA*, pages 684–697, 2010.