BULLETIN (New Series) OF THE
AMERICAN MATHEMATICAL SOCIETY
Volume 49, Number 1, January 2012, Pages 91–111
S 0273-0979(2011)01361-1
Article electronically published on November 7, 2011

ON KHOT'S UNIQUE GAMES CONJECTURE

LUCA TREVISAN

Abstract. In 2002, Subhash Khot formulated the *Unique Games Conjecture*, a conjecture about the computational complexity of certain optimization problems.

The conjecture has inspired a remarkable body of work, which has clarified the computational complexity of several optimization problems and the effectiveness of "semidefinite programming" convex relaxations.

In this paper, which assumes no prior knowledge of computational complexity, we describe the context and statement of the conjecture, and we discuss in some detail one specific line of work motivated by it.

1. Introduction

Khot formulated the *Unique Games Conjecture* in a very influential 2002 paper [23]. In the subsequent eight years, the conjecture has motivated and enabled a large body of work on the computational complexity of approximating combinatorial optimization problems (the original context of the conjecture) and on the quality of approximation provided by "semidefinite programming" convex relaxations (a somewhat unexpected by-product). Old and new questions in analysis, probability, and geometry have played a key role in this development.

Khot has recently written an excellent and broad survey paper on this body of work [24]. In this paper we take a complementary approach and, after providing the context and the statement of the conjecture, we focus mostly on one specific line of work, which is representative of the style of research in this area. The interested reader who would like to learn more about this research area is referred to Khot's paper [24].

2. The statement of the Unique Games Conjecture

In this section we give the context and the statement of the Unique Games Conjecture. The Unique Games Conjecture is a statement about the computational complexity of certain computational problems, so we begin with a very informal discussion of the computational complexity of "search problems", the notion of NP-completeness, and the application of NP-completeness to study the complexity of optimization problems. The reader can find a rigorous treatment in any introductory textbook on the theory of computation, such as Sipser [33]. We then explain the difficulties that arise in studying the complexity of approximations problems,

Received by the editors July 18, 2011, and, in revised form, July 25, 2011.
2010 *Mathematics Subject Classification.* Primary 68Q17.
This material is based upon work supported by the National Science Foundation under Grant No. CCF-1017403.

and introduce the notion of Probabilistically Checkable Proofs and the PCP Theorem. An excellent introductory treatment of these notions can be found in the textbook of Arora and Barak [5]. Finally, we state the Unique Games Conjecture as a variant of the PCP Theorem.

2.1. NP**-Completeness.** To briefly introduce computational complexity theory, consider the 3-*coloring* problem. In this computational problem, we are given as input an undirected graph[1] $G = (V, E)$, and the goal is to determine whether there is a *proper* 3-*coloring* of the vertices, that is, a function $c : V \to \{0, 1, 2\}$ such that $c(u) \neq c(v)$ for every $\{u, v\} \in E$. (If such proper colorings exist, we are also interested in finding one.)

The 3-coloring problem is easily solvable in finite time: just consider in some order all possible $3^{|V|}$ functions $c : V \to \{0, 1, 2\}$ and check each of them to see if it is a proper coloring. It is easy to improve the running time to about $2^{|V|}$, and there are non-trivial ways to achieve further speed-ups, but all the known algorithms have a worst-case running time that grows at a rate of $c^{|V|}$ for a constant $c > 1$, and they are unfeasible even on graphs with a few hundred vertices. Is there are an algorithm whose worst-case running time is bounded above by a polynomial function of $|V|$?

This is an open question equivalent to the P versus NP problem, one of the six unsolved Millenium Prize Problems. One thing we know, however, is that the 3-coloring problem is NP-*complete*, a notion introduced by Cook, Karp, and Levin [14, 28, 22]. Informally, NP is the collection of all computational problems that, like 3-coloring, involve searching an exponentially large list in order to find an element that satisfies a given property; P is the collection of all computational problems that can be solved by algorithms whose running time is bounded from above by a polynomial function of the input length.

A computational problem A *reduces* to a computational problem B if, informally, there is a way to "encode" instances of problem A as instances of problem B, so that the existence of polynomial-time algorithms for B automatically implies the existence of a polynomial-time algorithm for A.

A computational problem B is NP-*hard* if every problem $A \in NP$ reduces to B. If an NP-hard problem B is in NP, then B is said NP-*complete*. By the above discussion we see that an NP-complete problem can be solved in polynomial time if and only if $P = NP$.

It is not difficult to prove that NP-complete problems exist, and that simple and natural problems such as 3-coloring are NP-complete. The implications, however, are quite surprising when seen for the first time. For example, searching for a bounded-length proof of a given mathematical statement is an NP-problem, and so we have the following consequence of the NP-completeness of 3-coloring:

[1] An undirected graph is a pair (V, E) where V is a finite set and E is a set of unordered pairs of elements of V. The elements of V are called *vertices*, and the elements of E are called *edges*. The vertices u and v are called the *endpoints* of the edge $\{u, v\}$, and the edge $\{u, v\}$ is said to be *incident* on the vertices u and v.

Example 1. There is an algorithm that, given an integer n, runs in time polynomial in n and constructs a graph with $O(n^2)$ vertices such that the graph has a proper 3-coloring if and only if the Riemann Hypothesis has a proof of at most n pages.[2]

In fact, there is nothing special about the Riemann Hypothesis: for every mathematical statement S and every integer n it is possible to efficiently construct a graph of size polynomial in the length of S and in n such that the graph is 3-colorable if and only if S has a proof of at most n pages.

The consequences of $P = NP$, such as the ability to find mathematical proofs in time polynomial in the length of the proof, seem very implausible, and it is generally assumed that the resolution of the P versus NP question is that $P \neq NP$. Assuming $P \neq NP$ as a conjecture, then the proof that a problem is NP-complete can be seen as a conditional proof that the problem cannot be solved in polynomial time.

2.2. **Optimization problems and NP-completeness.** The theory of NP-completeness can also be used to reason about *combinatorial optimization problems*, that is problems in which one wants to pick from an exponentially long list an item that *maximizes* or *minimizes* a given cost function. In this paper we will mainly consider the following problems:

MAX CUT: In the Maximum Cut (abbreviated Max Cut) problem we are given as input an undirected graph $G = (V, E)$, and we want to find a bipartition $(S, V - S)$ of the set of vertices maximizing the number of edges that are *cut* by the partition, that is the number of edges that have one endpoint in S and one endpoint in $V - S$. Equivalently, we want to find a 2-coloring of the vertices that maximizes the number of edges that have endpoints of different colors.

Given a graph G and a number c, it is an NP-complete problem to determine if there is a solution that cuts at least c edges.

SPARSEST CUT: In the Sparsest Cut problem we are given a d-regular graph $G = (V, E)$ (that is, a graph in which each vertex is an endpoint of precisely d edges), and we again want to find a bipartition $(S, V - S)$, but this time we want to *minimize* the number of edges cut by the partition relative to how balanced the partition is. Namely, we want to find the partition that minimizes the ratio

$$\phi(S) := \frac{|E(S, V - S)|}{d \cdot |S| \cdot |V - S|/|V|},$$

where $E(S, V - S)$ is the set of cut edges. The normalization is chosen so that the ratio in the optimal solution is always between 0 and 1.

It is an NP-complete problem to decide, given a graph $G = (V, S)$ and a number r, whether there is a set $S \subseteq V$ such that $\phi(S) \leq r$.

Such NP-completeness results rule out (assuming $P \neq NP$) the possibility of algorithms of polynomial running time computing optimal solutions for the above problems. What about the computational complexity of finding *approximate* solutions?

[2]Here, and in subsequent examples, we are assuming that we have fixed a formal language in which to write mathematical proofs, and the length of the proof refers to the length of a fully formal proof in such a system.

2.3. **Approximation algorithms and PCPs.** The reductions that establish the above NP-completeness results do not offer much insight into the complexity of computing approximations. For example, the NP-completeness result for the Max Cut problem, relating it again to the task of finding a proof of the Riemann Hypothesis, gives the following implication:

Example 2. There is an algorithm that, given an integer n, runs in time polynomial in n and outputs a graph $G = (V, E)$ and a number c such that:

- If there is a proof of the Riemann Hypothesis of at most n pages, then there is a bipartition of V that cuts $\geq c$ edges;
- Every bipartition of G that cuts $\geq c$ edges can be efficiently converted to a valid proof of the Riemann Hypothesis of at most n pages.

Looking more carefully into the argument, however, one sees that the transformation has the following "robustness" property with respect to approximations:

Example 3. There is an algorithm that, given an integer n, runs in time polynomial in n and outputs a graph $G = (V, E)$ and a number c such that:

- If there is a proof of the Riemann Hypothesis of at most n pages, then there is a bipartition of V that cuts $\geq c$ edges;
- Every bipartition of G that cuts $\geq c - k$ edges can be efficiently converted to a valid proof of the Riemann Hypothesis of at most n pages with at most k mistakes.

This means that if we had, for example, an algorithm that finds in polynomial-time solutions to the Max Cut problem that are at most 1% worse than the optimal, we would have that we could find an n-page "proof" such that at most 1% of the steps are wrong. Since it is always easy to come up with a proof that contains at most one mistake ("trivially, we have $0 = 1$, hence, ..."), this doesn't cause any contradiction.

This does not mean that approximating the Max Cut problem is easy: it just means that the instances produced by the NP-completeness proof are easy to approximate, and if one wants to prove a statement of the form "if there is a polynomial-time algorithm for the Max Cut problem that finds solutions at most 1% worse than the optimum, then $P = NP$", then such a result requires reductions of a rather different form from the ones employed in the classical theory of NP-completeness.

Indeed, with few exceptions, proving intractability results for approximation problems remained an open question for two decades, until the proof of the *PCP Theorem* in the early 1990s by Arora, Lund, Motwani, Safra, Sudan, and Szegedy [4, 3]. The PCP Theorem (PCP stands for *Probabilistically Checkable Proofs*) can be thought of as describing a format for writing mathematical proofs such that even a "proof" in which up to, say, 1% of the steps are erroneous implies the validity of the statement that it is supposed to prove.

Theorem 1 (The PCP Theorem)**.** *There is a constant ϵ_0 and a polynomial-time algorithm that on input a graph $G = (V, E)$ outputs a graph $G' = (V', E')$ such that*

- *If G has a proper 3-coloring, then so does G'.*
- *If there is a coloring $c' : V' \to \{1, 2, 3\}$ such that at least a $1 - \epsilon_0$ fraction of the edges of G' are properly colored by G', then G has a proper 3-coloring, and a proper 3-coloring can be efficiently constructed from c'.*

The contrapositive of the second property is that if G is not a 3-colorable graph, then G' is a graph that is *not even approximately* 3-colorable, that is, G' is a graph such that, in every 3-coloring of the vertices, at least an ϵ_0 fraction of the edges have endpoints of the same color.

To see how this leads to probabilistically checkable proofs, let us return to our running example of whether, for a given n, there is an n-page proof of the Riemann Hypothesis. For a given n, we can construct in time polynomial in n a graph G such that an n-page exists if and only if there is a proper 3-coloring of G. From G we can construct, again in time polynomial in n, a graph G' as in the PCP Theorem. Now, an n-page proof of the Riemann Hypothesis can be encoded (at the cost of a polynomial blow-up in size) as a proper coloring of G'. Given a candidate proof, presented as a coloring of G', we can think of it as having $|E'|$ "steps", each being the verification that one of the edges of G' has indeed endpoints of different colors. If an n-page proof of the Riemann Hypothesis exists, then there is a proof, in this format, all of whose steps are correct; if there is no n-page proof of the Riemann Hypothesis, however, every "proof" is now such that at least an ϵ_0 fraction of the steps are wrong. If we sample at random $100/\epsilon_0$ edges of G', and check the validity of the given coloring just on those edges, we will find a mistake with extremely high probability. Thus the PCP Theorem gives a way to write down mathematical proofs, and a probabilistic verification procedure to check the validity of alleged proofs that *reads only a constant number of bits of the proof* and such that valid proofs pass the probabilistic test with probability 1, and if the test passes with probability higher than $(1-\epsilon_0)^{100/\epsilon_0} \approx e^{-100}$, then a valid proof exists.

This application to proof checking is impractical for a series of reasons, and we are describing it as a way to gain intuition about the statement of the PCP Theorem, but similar encodings might have future applications to the design of cryptographic protocols. In any case, the main application and motivation of the PCP Theorem is the study of the complexity of finding approximations to combinatorial optimization problems.

2.4. **Label Cover.** Various forms of the PCP Theorems are known, which are tailored to the study of specific optimization problems. A very versatile form of the theorem, which was proved by Ran Raz [31] (solving a question raised by the work of Bellare et al. [10, 9]), refers to the *Label Cover* problem.

Definition 2 (Label Cover)**.** An input to the Label Cover problem with range Σ (Σ is a finite set) is a set of equations of the form

$$X_i = \sigma_{i,j}(Y_j),$$

where $\sigma_{i,j} : \Sigma \to \Sigma$ are functions specified as part of the input.

The goal is to find an assignment to the variables X_i and Y_j that satisfies as many equations as possible.

For example, the following is an instance of Label Cover with range $\mathbb{Z}/5\mathbb{Z}$:

$$\begin{aligned} X_1 &= Y_1^2 - 1 \bmod 5, \\ X_2 &= Y_1 - 1 \bmod 5, \\ X_1 &= Y_2^4 + 1 \bmod 5. \end{aligned}$$

The first and third equations are not simultaneously satisfiable, and so an optimal solution to the above instance is to satisfy two of the equations, for example the first and the second with the assignment $X_1 := 4$, $X_2 := 4$, $Y_1 := 0$, $Y_2 := 0$.

Notice that while the example above involved equations of an algebraic nature, it is possible to use any function in the definition of an instance of the Label Cover problem.

Theorem 3 (Raz [31]). *For every $\epsilon > 0$, there is a Σ, $|\Sigma| \leq 1/\epsilon^{O(1)}$ and a polynomial-time algorithm that on input a graph G outputs an instance C of Label Cover with range Σ such that the following hold:*

- *If G has a proper 3-coloring, then in C there is an assignment to the variables that satisfies all constraints;*
- *If G is not properly 3-colorable, then every assignment to the variables of C satisfies at most an ϵ fraction of the equations.*

This form of the PCP Theorem is particularly well suited as a starting point for reductions, because in the second case we have the very strong guarantee that it is impossible to satisfy even just an ϵ fraction of the equation. For technical reasons, it is also very useful that each equation involves only two variables.

The approach to derive intractability, for example for a graph problem, from Theorem 3 is to encode each variable as a small graph, and to lay out edges in such a way that the only way to have a good solution in the graph problem is to have it so that it defines a good solution for the Label Cover problem. If we are studying a cut problem, for example, and we have collection of vertices $v_{X,1}, \ldots, v_{X,k}$ corresponding to each variable X in the Label Cover instance, then a cut $(S, V - S)$ in the graph gives a k-bit string $(b_{X,1}, \ldots, b_{X,k})$ for every variable X of Label Cover, corresponding to which of the k vertices does or does not belong to S.

The problem then becomes the following:

(1) To make sure that only bit strings close to a valid code word can occur in a near-optimal solution;
(2) To make sure that in near optimal solutions the decodings satisfy a large number of equations.

Task (2) is typically much harder than task (1), especially in reductions to graph problems. Indeed most NP-completeness results for approximating graph optimization problems have proceeded by first reducing Label Cover to an intermediate simpler problem, and then reducing the intermediate problem to the graph problem, but at the cost of weaker intractability results than the conjectured ones.

In 2002, Khot [23] formulated a conjecture that considerably simplifies (2), essentially making it of difficulty comparable to (1).

2.5. The Unique Games Conjecture.

Definition 4 (Unique Game). A *unique game* with range Σ is a set of equations of the form

$$X_i = \sigma_{i,j}(Y_j),$$

where $\sigma_{i,j} : \Sigma \to \Sigma$ are *bijective* functions specified as part of the input.

The goal is to find an assignment to the variables X_i and Y_j that satisfies as many equations as possible.

For example, the following is a unique game with range $\mathbb{Z}/5\mathbb{Z}$:

$$\begin{aligned} X_1 &= Y_1 + 3 \bmod 5, \\ X_2 &= Y_1 + 1 \bmod 5, \\ X_1 &= Y_2 - 1 \bmod 5, \\ X_2 &= Y_2 - 1 \bmod 5. \end{aligned}$$

In the above example, it is not possible to satisfy all four equations, but the optimal solution $X_1 := 3, X_2 := 1, Y_1 := 0, Y_2 = 2$ satisfies three of the equations.

Notice that the only difference between a Label Cover instance and a Unique Game is that, in a Unique Game, the functions that define the equations have to be bijective. This is, however, a substantial difference.

In particular, given a Unique Game that has a solution that satisfies all equations, such a solution can be found very quickly in time linear in the number of equations, while in a satisfiable Label Cover instance it is an NP-hard problem to even find a solution that satisfies a small fraction of equations.

But what if we are given a Unique Game in which there is a solution that satisfies, say, a 99% fraction of the equation?

Conjecture 1 (Unique Games Intractability Conjecture)**.** For every $1/2 > \epsilon > 0$, there is a Σ such that there is no polynomial-time algorithm that, given an instance of Unique Games with range Σ in which it is possible to satisfy at least a $1 - \epsilon$ fraction of equations, finds a solution that satisfies at least an ϵ fraction of equations.

If $P = NP$, then Conjecture 1 is false; this means that proving Conjecture 1 would require first proving $P \neq NP$, which is beyond the reach of current techniques. The strongest evidence that we can currently hope to prove in favor of Conjecture 1 is:

Conjecture 2 (Unique Games NP-Hardness Conjecture)**.** For every $1/2 > \epsilon > 0$, there is a Σ and a polynomial-time algorithm that, on input, a graph G outputs a unique games instance U with range Σ, such that

- If G is properly 3-colorable, then there is an assignment that satisfies at least a $1 - \epsilon$ fraction of equations in U;
- If G is not properly 3-colorable, then every assignment to the variables of U satisfies at most an ϵ fraction of equations.

If Conjecture 2 is true, then every inapproximability result proved via a reduction from Unique Games establishes an NP-hardness of approximation, in the same way as a reduction starting from Label Cover.

2.5.1. *Consequence for Max Cut.* In the following we let

$$\alpha_{GW} := \min_{1/2<\rho<1} \frac{\frac{1}{\pi} \cdot \arccos 1 - 2\rho}{\rho} \approx 0.878567. \tag{1}$$

And we let ρ_{GW} be the value of ρ that minimizes the above expression. The above constant comes up in the remarkable algorithm of Goemans and Williamson [20].

Theorem 5 (Goemans and Williamson [20])**.** *There is a polynomial-time algorithm that, given as input a graph $G = (V, E)$, finds a bipartition that cuts at least $\alpha_{GW} \cdot opt$ edges, where opt is the number of edges cut by an optimal bipartition of G. Furthermore, if $opt/|E| = 1 - \epsilon \geq \rho_{GW}$, then the bipartition found by the algorithm cuts at least*

$$\frac{1}{\pi} \cdot \arccos(-1 + 2\epsilon) \cdot |E| \tag{2}$$

edges.

The value of expression (2) is approximately

$$\left(1 - \frac{2}{\pi}\sqrt{\epsilon}\right) \cdot |E|$$

when ϵ is small.

It is known that the existence of a polynomial-time algorithm for Max Cut with approximation ratio better than 16/17 implies that $P = NP$ [34, 21], but no NP-hardness result is known in the range between $\alpha \approx .878$ and $16/17 \approx .941$, and there has been no progress on this problem since 1997.

Work of Khot, Kindler, Mossel, and O'Donnel [25], together with later work of Mossel, O'Donnel, and Oleszkiewicz [29], proves that no improvement is possible over the Goemans–Williamson algorithm assuming the Unique Games Intractability Conjecture.

Theorem 6 ([25, 29])**.** *Suppose that there is a $\delta > 0$, a $\rho > 0$, and a polynomial-time algorithm that given a graph $G = (V, E)$ in which an optimal cut cuts $\rho \cdot |E|$ vertices finds a solution that cuts at least $\frac{1}{\pi} \cdot (\arccos(1 - 2\rho) + \delta) \cdot |E|$ edges. Then the Unique Games Intractability Conjecture is false.*

In particular, by taking $\rho = \rho_{GW}$ we have that, for every $\delta > 0$, the existence of a polynomial-time algorithm that, on input, a graph in which the optimum is c finds a solution that cuts more than $(\alpha_{GW} + \delta) \cdot c$ edges would contradict the Unique Games Intractability Conjecture. So, assuming the conjecture, the constant α_{GW} is precisely the best achievable ratio between the value of polynomial-time constructible solutions and optimal solutions in the Max Cut problem.

In Section 3 we will present an overview of the proof of Theorem 6.

2.5.2. *Consequence for Sparsest Cut.* The algorithm achieving the best ratio between the quality of an optimal solution and the quality of the solution found in polynomial time is due to Arora, Rao, and Vazirani [8].

Theorem 7 ([8])**.** *There is a polynomial-time algorithm that, given a graph $G = (V, E)$, finds a set C such that*

$$\phi(C) \leq O(\sqrt{\log |V|}) \cdot \phi(C^*),$$

where C^* is an optimal solution to the Sparsest Cut problem.

A classical algorithm based on spectral graph theory achieves a better approximation in graphs in which the optimum is large.

Theorem 8 (Spectral Partitioning [1, 2])**.** *There is a polynomial-time algorithm that, given a graph $G = (V, E)$, finds a set C such that*

$$\phi(C) \leq O(\sqrt{\phi(C^*)}),$$

where C^* is an optimal solution to the Sparsest Cut problem.

Theorem 9 ([25, 29])**.** *There is an absolute constant $c > 0$ such that the following is true.*

Suppose that there is a $\delta > 0$, an $\epsilon > 0$, and a polynomial-time algorithm that, given a graph $G = (V, E)$ in which the Sparsest Cut C^* satisfies $\phi(C^*) \leq \epsilon$, finds a cut C such that

$$\phi(C) \leq c \cdot \sqrt{\epsilon} - \delta;$$

then the Unique Games Intractability Conjecture is false.

In particular, assuming the conjecture, the trade-off between optimum and approximation in the Spectral Partitioning algorithm cannot be improved, and the approximation ratio in the Arora–Rao–Vazirani algorithm cannot be improved to a constant.

3. The Maximum Cut problem

A general approach to reduce Unique Games (and, in general, Label Cover) with range Σ to other problems is to ensure that a solution in the target problem associates to each variable X of the unique game a function $f_X : \{-1,1\}^\Sigma \to \{-1,1\}$. Then we define a way to "decode" a function $f_X : \{-1,1\}^\Sigma \to \{-1,1\}$ to a value $a_X \in \Sigma$, and we aim to prove that if we have a good solution in the target problem, then the assignment $X := a_X$ to each variable X defines a good solution in the Unique Games instance. The general idea is that if a function "essentially depends" upon one of its variables, then we decode it to the index of the variable that it depends upon.

3.1. **The reduction from Unique Games to Max Cut.** We outline this method by discussing the proof of Theorem 6. To prove the theorem, we start from a Unique Games instance U with range Σ such that a $1-\epsilon'$ fraction of equations can be satisfied. We show how to use the assumption of the theorem to find a solution that satisfies at least an ϵ' fraction of equations. We do so by constructing a graph G, applying the algorithm to find a good approximation to Max Cut in the graph, and then converting the cut into a good solution for the Unique Games instance.

If U has N variables, then G has $N \cdot 2^\Sigma$ vertices, a vertex $v_{X,a}$ for every variable X of U, and for every value $a \in \{0,1\}^\Sigma$.

We define G as a *weighted* graph, that is a graph in which edges have a positive real-value weight. In such a case, the value of a cut is the total weight (rather than the number) of edges that are cut. There is a known reduction from Max Cut in a weighted graph to Max Cut in unweighted simple graphs [15], so there is no loss of generality in working with weights.

We introduce the following action of the symmetric group of Σ on the vertices of G. If $x \in \{-1,1\}^\Sigma$ is a vector of $|\Sigma|$ bits indexed by the elements of Σ, and $\sigma : \Sigma \to \Sigma$ is a bijection, we denote by $x \circ \sigma$ the vector $x \circ \sigma \in \{0,1\}^\Sigma$ such that $(x \circ \sigma)_i := x_{\sigma(i)}$.

We also define the *noise operator* N_ρ as follows: if $x \in \{0,1\}^\Sigma$ is a boolean vector, then $N_\rho(x)$ is the random variable generated by changing each coordinate of x independently with probability ρ, and leaving it unchanged with probability $1-\rho$.

The edge set of G is defined so that its total weight is 1, and we describe it as a probability distribution:

- Pick two random equations $X = \sigma(Y)$ and $X = \sigma'(Y')$ in U conditioned on having the same left-hand side.
- Pick a random element $a \in \{0,1\}^\Sigma$, and pick an element $b \in N_\rho(a)$.
- Generate the edge $(v_{Y,a\circ\sigma}, v_{Y',b\circ\sigma'})$.

Let A be an optimal assignment for the Unique Games instance U. Consider the cut of G in which $S = \{v_{Y,a} : a_{A(Y)} = 1\}$. This vertex bipartition cuts edges of total weight at least $\rho - 2\epsilon'$. From our assumption, we can find in polynomial time a cut S that cuts a $\frac{1}{\pi} \cdot (\arccos 1 - 2\rho) + \delta$ fraction of edges. We want to show how

to extract from S an assignment for the Unique Games that satisfies a reasonably large number of equations.

First we note that S assigns a bit to each variable X and to each $a \in \{-1,1\}^\Sigma$. Let us call

$$f_Y(a) = 1 \text{ if } a \in S$$

and

$$f_Y(a) = -1 \text{ if } a \notin S.$$

We want to decode each of these functions $f_X : \{-1,1\}^\Sigma \to \{-1,1\}$ into an index $i \in \Sigma$. We describe a probabilistic decoding process $Dec(\cdot)$ later.

Some calculations show that the functions we derive in such a way have the property

$$\mathop{\mathbb{E}}_{X,Y,Y',a,b}[f_Y(a \circ \sigma) \neq f_{Y'}(b \circ \sigma')] \geq \frac{1}{\pi} \cdot (\arccos 1 - 2\rho) + \delta,$$

and from this we want to derive

$$\mathop{\mathbb{E}}_{X,Y,Y'}[\sigma(Dec(f_Y)) = \sigma'(Dec(f_{Y'}))] \geq \Omega_{\rho,\delta}(1),$$

from which it is easy to see that from the decodings $Dec(f_Y)$ we can reconstruct an assignment for all variables which satisfies at least an ϵ' fraction of equations in the unique game.

Some manipulations show that, essentially, it is sufficient to prove the following lemma:

Lemma 10 (Main Lemma)**.** *There is a probabilistic symmetric algorithm $Dec(\cdot)$ that on input a function $f : \{-1,1\}^\Sigma \to \{-1,1\}$ outputs an element $i \in \Sigma$, and such that the following is true.*

Suppose that $f : \{-1,1\}^\Sigma \to \{-1,1\}$ is such that

$$\mathbb{P}[f(x) \neq f(N_\rho x)] \geq \frac{1}{\pi} \cdot \arccos(1-2\rho) + \delta. \tag{3}$$

Then there is an index $i \in \Sigma$ such that

$$\mathbb{P}[Dec(f) = i] \geq \Omega_{\delta,\rho}(1).$$

We say that the decoding is *symmetric* if the distribution of $Dec(f(\sigma(\cdot)))$ is the same as the distribution $\sigma(Dec(f(\cdot)))$ for every bijection $\sigma : \Sigma \to \Sigma$.

(Technically, the Main Lemma is not sufficient as stated. An extension that deals with all bounded real-valued functions is necessary. The boolean case, which is simpler to state and visualize, captures all the technical difficulties of the general case.)

3.2. **The proof of the Main Lemma.** Before discussing the proof of the Main Lemma, we show that it is tight, in the sense that from a weaker assumption in equation (3) it is not possible to recover the conclusion.

Consider the majority function $Maj : \{-1,1\}^\Sigma \to \{-1,1\}$ such that $Maj(x) = 1$ if and only if x has at least $|\Sigma|/2$ ones. (That is, $Maj(x) := sign(\sum_i x_i)$.) Then Maj is a *symmetric* function, in the sense that $Maj(x \circ \sigma) = Maj(x)$ for every bijection σ. This implies that for every symmetric decoding algorithm Dec we have that $Dec(Maj)$ is the uniform distribution over Σ, and so every index i has

probability $1/|\Sigma|$ which goes to zero even when the other parameters in the Main Lemma are fixed. A standard calculation shows that, for large Σ,

$$\mathbb{E}[Maj(x) \neq Maj(N_\rho(x))] \approx \frac{1}{\pi} \arccos(1-2\rho),$$

so we have an example in which equation (3) is nearly satisfied but the conclusion of the Main Lemma fails.

This example suggests that, if the Main Lemma is true, then the functions that satisfy equation (3) must be non-symmetric, that is, they must not depend equally on all the input variables, and the decoding procedure $Dec(\cdot)$ must pick up certain input variables that the function depends upon in a special way.

Another example to consider is that of the functions arising in the bipartitions that are derived from an optimal solution in the unique game instance U. In that case, for every variable Y, the corresponding function f_Y is of the form $f_Y(x) := x_i$ where i is the value assigned to Y in the optimal solution. In this case, we would expect the decoding algorithm to output the index i. In general, if f depends only on a small number of variables, we would expect Dec to only output the indices of those variables.

These observations suggest the use of the notion of *influence* of input variables. If $f : \{-1,1\}^\Sigma \to \{-1,1\}$ is a boolean function, then the influence of variable $i \in \Sigma$ on f is the probability

$$\mathrm{Inf}_i(f) := \mathop{\mathbb{P}}_{x \in \{0,1\}^\Sigma}[f(x_1,\ldots,x_k) \neq f(x_1,\ldots,x_{i-1},-x_i,x_{i+1},\ldots,x_k)],$$

where we identified, for simplicity, Σ with $\{1,\ldots,k\}$, where $k := |\Sigma|$.

It is natural to consider the decoding algorithm that picks an index i with probability proportional to $\mathrm{Inf}_i(f)$; note that this process is symmetric.

There is, unfortunately, a counterexample. Consider the function

$$f(x_1,\ldots,x_k) := Maj(x_1, x_2, Maj(x_3,\ldots,x_k)),$$

and take $\rho = 1-\epsilon$. Then $\frac{1}{\pi} \cdot \arccos(1-2\rho) \approx 1 - \frac{2}{\pi}\sqrt{\epsilon}$, and one can compute that

$$\mathbb{P}[f(x) \neq f(N_\rho(x))] \approx 1 - \epsilon - \frac{1}{\pi}\sqrt{\epsilon} > 1 - \frac{2}{\pi}\sqrt{\epsilon} + \Omega_\epsilon(1).$$

This means that we expect the decoding algorithm to select some index with a probability that is at least a fixed constant for every fixed ϵ.

When we compute the influence of the variables of f, however, we find out that x_1 and x_2 have constant influence $1/2$, while the variables $x_3,\ldots,x_k$ have influence approximately $1/\sqrt{k}$. This means that the sum of the influences is about $\sqrt{k}$, and so x_1 and x_2 would be selected with probability about $1/\sqrt{k}$, and the remaining variables with probability about $1/n$. In particular, all probabilities go to zero with $k = |\Sigma|$, and so a decoding algorithm based only on influence does not satisfy the conditions of the Main Lemma.

In order to introduce the correct definition, it helps to introduce discrete Fourier analysis over the Hamming cube. For our purposes, only the following facts will be used. One is that if $g : \{-1,1\}^\Sigma \to \mathbb{R}$ is a real-valued function, then there is a unique set of real values $\hat{g}(S)$, one for each subset $S \subseteq \Sigma$, such that

$$g(x) = \sum_S \hat{g}(S) \cdot \prod_{i \in S} x_i.$$

The values $\hat{g}(S)$ are called the Fourier coefficients of g.

In particular, if $f : \{-1,1\}^\Sigma \to \{-1,1\}$ is a boolean function, then $\sum_S \hat{f}^2(S) = 1$. It is easy to see that

$$\mathrm{Inf}_i(f) = \sum_{S : i \in S} \hat{f}^2(S).$$

The fact that $\sum_i \hat{f}^2(S) = 1$ suggests that $\hat{f}$ naturally defines a probability distribution. Unfortunately, it is a probability distribution over subsets of Σ, rather than a probability distribution over *elements* of Σ. A natural step is to consider the algorithm *Dec* defined as follows: sample a set $S \subseteq \Sigma$ with probability equal to $\hat{f}^2(S)$, then output a random element of S. In particular, we have

$$\mathbb{P}[Dec(f) = i] = \sum_{S : i \in S} \frac{\hat{f}^2(S)}{|S|}, \tag{4}$$

which is similar to the expression for the influence of i, but weighted to give more emphasis to the Fourier coefficients corresponding to smaller sets.[3]

If we go back to the function $Maj(x_1, x_2, Maj(x_3, \ldots, x_k))$, we see that the algorithm defined in (4) has a probability of generating x_1 and x_2 which is at least an absolute constant, and which does not go to zero with k.

The decoding algorithm described in equation (4) turns out to be the correct one. Proving the Main Lemma reduces now to proving the following result.

Lemma 11 (Main Lemma—Restated)**.** *Suppose that $f : \{0,1\}^\Sigma \to \{0,1\}$ is such that*

$$\mathbb{P}[f(x) \neq f(N_\rho x)] \geq \frac{1}{\pi} \cdot \arccos(1 - 2\rho) + \delta. \tag{5}$$

Then there is an index $i \in \Sigma$ such that

$$\sum_{S : i \in S} \frac{\hat{f}^2(S)}{|S|} \geq \Omega_{\delta,\rho}(1).$$

The proof has two parts:

- An *invariance theorem*, due to Mossel, O'Donnell, and Oleszkiewicz [29], showing that the Main Lemma is true in the boolean setting provided that a "Gaussian version" of the Lemma holds for functions taking real inputs with Gaussian distribution is true;
- A theorem of Borell [11] establishing the Gaussian version of the lemma.

3.3. **The Invariance Theorem and Borell's theorem.** A starting point to gain intuition about the Invariance Theorem is to consider the Central Limit Theorem. Suppose that $X_1, \ldots, X_n$ is a collection of independent boolean random variables, each uniform over $\{-1,1\}$, and suppose that $a_1, \ldots, a_n$ are arbitrary coefficients. Then the random variable

$$\sum_i a_i X_i$$

is going to be close to a Gaussian distribution of average zero and variance $\sum_i a_i^2$, provided that the coefficients are reasonably smooth. (It is enough that if we scale them so that $\sum_i a_i^2 = 1$, then $\sum_i a_i^3$ is small.)

[3] An important point is that with probability $\hat{f}(\emptyset)^2$ we generate the empty set, and the operation of "selecting a random element" of the empty set is undefined. In such a case, the decoding algorithm outputs a special failure symbol $\perp$ not in Σ.

Suppose now that, instead of considering a sum, that is, a degree-1 function, we take an n-variate, low-degree polynomial p and we consider the random variable

$$p(X_1, \ldots, X_n).$$

We cannot say any more that it has a distribution close to a Gaussian and, in fact, it does not seem that we can say anything at all. Looking back at the Central Limit Theorem, however, we can note that the "right" way of formulating it is to consider a collection $X_1, \ldots, X_n$ of independent boolean random variables each uniform over $\{-1, 1\}$, and also a collection of independent Gaussian random variables $Z_1, \ldots, Z_n$ each with mean zero and variance 1. Then we have that the two random variables

$$\sum_i a_i X_i \text{ and } \sum_i a_i Z_i$$

are close provided that the a_i are smooth.

This is exactly the same statement as before, because the distribution $\sum_i a_i Z_i$ happens to be a Gaussian distribution of mean zero and variance $\sum_i a_i^2$.

This formulation, however, is a natural analog to the case of low-degree polynomials. The Invariance Theorem states that if p is a sufficiently "smooth" low degree polynomial, then the random variables

$$p(X_1, \ldots, X_n) \text{ and } p(Z_1, \ldots, Z_n)$$

are close. A result of this nature was first proved by Rotar [32].

When we apply the Invariance Theorem to a smoothed and truncated version of the Fourier transform of the function f in the Main Lemma, we have that either such a function is a "smooth polynomial" to which the Invariance Theorem applies, or else the conclusion holds and there is a coordinate with noticeably high probability of being output by the decoding algorithm. If the Invariance Theorem applies, then the probability that f changes value on anti-correlated boolean inputs is approximately the probability that a function changes its value on anti-correlated Gaussian inputs. The latter is given by a theorem of Borell.

Theorem 12 (Borell)**.** *Suppose $f : \mathbb{R}^n \to [-1, 1]$ is a measurable function according to the standard Gaussian measure in $\mathbb{R}^n$ and such that $\mathbb{E} f = 0$. For an element $x \in \mathbb{R}^n$ and for $0 \leq \rho \leq 1/2$, let $N_\rho(x)$ be the random variable $(1 - 2\rho) \cdot x + \sqrt{1 - (1 - 2\rho)^2} z$ where z is a standard Gaussian random variable. Then*

$$\mathbb{P}[f(x) \neq f(N_\rho(x))] \geq \frac{1}{\pi} \arccos(1 - 2\rho).$$

There are a few ways in which Borell's theorem is not the "Gaussian analog" of the Main Lemma. Notably, there is a condition on the expectation of f , there is a lower bound, rather than an upper bound, to the probability that f changes value, and the theorem applies to the range $\rho \in [0, 1/2]$, while we are interested in the "anti-correlation" case of $\rho \in [1/2, 1]$. There is a simple trick (consider only the "odd part" of the Fourier expansion of the boolean function f—that is only the terms corresponding to sets S of odd size) that takes care of all these differences.

3.4. **How did we use the Unique Games Conjecture?** When we stated the Unique Games Conjecture, we made the following informal claim, here rephrased in abbreviated form:

To reduce Label Cover to a graph optimization problem like Max Cut, we map variables to collections of vertices and we map equations to collections of edges; then we show how to "encode" assignments to variables as 2-colorings of vertices which cut a $\geq c_1$ fraction of edges, and finally (this is the hardest part of the argument) we show that given a 2-coloring that cuts a $\geq c_2$ fraction of edges, then

(1) *The given 2-coloring must be somewhat "close" to a 2-coloring coming from the encoding of an assignment; and*
(2) *If we "decode" the given 2-coloring to an assignment to the variables, such an assignment satisfies a noticeable fraction of equations.*

Starting our reduction from a Unique Game instead of a Label Cover problem, we need only prove (1) above, and (2) more or less follows for free.

To verify this claim, we "axiomatize" the properties of a reduction that only achieves (1): we describe a reduction mapping a single variable to a graph, such that assignments to the variable are mapped to good cuts, and somewhat good cuts can be mapped back to assignments to the variable. The reader can then go back to our analysis of the Max Cut inapproximability proof in the previous post, and see that almost all the work went into establishing the existence of a family of graphs satisfying the properties below.

Definition 13 ((c_1, c_2)-graph family)**.** A (c_1, c_2)-graph family is a collection of graphs $G_m = (V_m, E_m)$, for each positive integer m, together with an encoding function $Enc_m : \{1, \ldots, m\} \to 2^{V_m}$ and a randomized decoding process $Dec_m : 2^{V_m} \to \{1, \ldots, m\}$ such that the following hold:

- For every m and every $i \in m$, let $S_i := Enc_m(i)$. Then the partition $(S_i, V_m - S_i)$ cuts at least a c_1 fraction of the edges of G_m.
- If $(S, V_m - S)$ is a partition of the vertices of G_m that cuts at least a $c_2 + \delta$ fraction of the edges, then there is an index $i \in \{1, \ldots, m\}$ such that the probability
$$\mathbb{P}[Dec_m(S) = i] \geq p(\delta) > 0$$
is at least a positive quantity $p(\delta)$ independent of m.
- The encoding and decoding procedures are *symmetric*. That is, it is possible to define an action of the symmetric group of $\{1, \ldots, m\}$ on V_m such that for every $i \in m$ and every bijection $\sigma : \{1, \ldots, m\} \to \{1, \ldots, m\}$ we have
$$Enc_m(\sigma(i)) = \sigma(Enc_m(i))$$
and
$$Dec_m(\sigma(S)) \approx \sigma(Dec_m(S)),$$
where $D_1 \approx D_2$ means that D_1 and D_2 have the same distribution, and $\sigma(S) := \{x \circ \sigma : x \in S\}$, where $x \circ \sigma$ is the action of σ on x.

We claim that, in the previous post, we defined a $(1 - \epsilon, 1 - \frac{2}{\pi}\sqrt{\epsilon})$-graph family. The graph family is the following. For a given m, the following hold:

(1) The vertex set is $V_m := \{-1, 1\}^m$.

(2) The graph is a weighted complete graph with edges of total weight 1. The weight of edge (x, y) is the probability of generating the pair (x, y) by sampling x at random and sampling y from the distribution $N_{1-\epsilon}(x)$.
(3) $Enc_m(i)$ defines the bipartition $(S_i, V_m - S_i)$ in which S_i is the set of all vertices x such that $x_i = 1$.
(4) $Dec_m(S)$ proceeds as follows. Define $f(x) := -1$ if $x \in S$ and $f(x) := 1$ if $x \notin S$. Compute the Fourier expansion

$$f(x) = \sum_R \hat{f}(R)(-1)^{\sum_{i \in R} x_i}.$$

Sample a set R with probability proportional to $\hat{f}^2(R)$, and then output a random element of R.

4. Semidefinite programming and Unique Games

Solving an instance of a combinatorial optimization problem of minimization type is a task of the form

$$\begin{array}{l} \max cost(z) \\ \text{subject to} \\ z \in Sol \end{array} \tag{6}$$

where Sol is the set of admissible solutions and $cost(z)$ is the cost of solution z. For example the problem of finding the maximum cut in a graph $G = (V, E)$ is a problem of the above type where Sol is the collection of all subsets $S \subseteq V$, and $cost(S)$ is the number of edges cut by the vertex partition $(S, V - S)$.

If $Sol \subseteq Rel$, and $cost' : Rel \to \mathbb{R}$ is a function that agrees with $cost()$ on Sol, then we call the problem

$$\begin{array}{l} \max cost'(z) \\ \text{subject to} \\ z \in Rel \end{array} \tag{7}$$

a *relaxation* of the problem in (6). The interest in this notion is that combinatorial optimization problems in which the solution space is discrete are often NP-hard, while there are general classes of optimization problems defined over a continuous *convex* solution space that can be solved in polynomial time. A fruitful approach to approximating combinatorial optimization problems is thus to consider relaxations to tractable convex optimization problems, and then argue that the optimum of the relaxation is close to the optimum of the original discrete problem. See the book of Vazirani [35] for several appications of this approach.

The Unique Games Intractability Conjecture is deeply related to the approximation quality of *semidefinite programming* relaxations of combinatorial optimization problems.

4.1. Semidefinite programming.

A symmetric matrix A is positive semidefinite, written $A \succeq \mathbf{0}$, if all its eigenvalues are non-negative. We write $A \succeq B$ if $A - B$ if positive semidefinite. We quote without proof the following facts:

- A matrix $A \in \mathbb{R}^{n \times n}$ is positive semidefinite if and only if there are vectors $v^1, \ldots, v^n \in \mathbb{R}^m$ such that for every i, j we have $A_{ij} = \langle v^i, v^j \rangle$. Furthermore, there is an algorithm of running time polynomial in n that, given a

matrix A, tests whether A is positive semidefinite and, if so, finds vectors $v^1, \ldots, v^n$ as above.

- The set of positive semidefinite matrices is a convex subset of $\mathbb{R}^{n \times n}$. More generally, it is a *convex cone*, that is, for every two positive semidefinite matrices A, B and non-negative scalars α, β, the matrix $\alpha A + \beta B$ is positive semidefinite.

It is often the case that the optimizing a linear function over a convex subset of $\mathbb{R}^N$ is a polynomial-time solvable problem, and indeed there are polynomial-time algorithms for the following problem:

Definition 14 (Semidefinite Programming)**.** The Semidefinite Programming problem is the following computational program: given matrices $C, A^1, \ldots, A^m \in \mathbb{R}^{n \times n}$ and scalars $b_1, \ldots, b_m \in \mathbb{R}$, find a matrix X that solves the following optimization problem (called a semidefinite program):

$$\begin{array}{l} \max C \bullet X \\ \text{subject to} \\ A^1 \bullet X \leq b_1 \\ A^2 \bullet X \leq b_2 \\ \cdots \\ A^m \bullet X \leq b_m \\ X \succeq \mathbf{0} \end{array} \tag{8}$$

where we use the notation $A \bullet B := \sum_{ij} A_{ij} \cdot B_{ij}$.

In light of the characterization of positive semidefinite matrices described above, the semidefinite program (8) can be written equivalently as follows:

$$\begin{array}{l} \max \sum_{ij} C_{ij} \cdot \langle v^i, v^j \rangle \\ \text{subject to} \\ \sum_{ij} A^1_{ij} \cdot \langle v^i, v^j \rangle \leq b_1 \\ \sum_{ij} A^2_{ij} \cdot \langle v^i, v^j \rangle \leq b_2 \\ \cdots \\ \sum_{ij} A^m_{ij} \cdot \langle v^i, v^j \rangle \leq b_m \\ v^1, \ldots, v^n \in \mathbb{R}^n. \end{array} \tag{9}$$

That is, as an optimization problem in which we are looking for a collection $v^1, \ldots, v^n$ of vectors that optimize a linear function of their inner products subject to linear inequalities about their inner products.

4.2. **Semidefinite Programming and approximation algorithms.** A *quadratic program* is an optimization problem in which we are looking for reals $x_1, \ldots, x_n$ that optimize a quadratic form subject to quadratic inequalities, that is an optimization problem that can be written as follows:

$$\begin{array}{l} \max \sum_{ij} C_{ij} \cdot x_i \cdot x_j \\ \text{subject to} \\ \sum_{ij} A^1_{ij} \cdot x_i \cdot x_j \leq b_1 \\ \sum_{ij} A^2_{ij} \cdot x_i \cdot x_j \leq b_2 \\ \cdots \\ \sum_{ij} A^m_{ij} x_i \cdot x_j \leq b_m \\ x_1, \ldots, x_n \in \mathbb{R}. \end{array} \tag{10}$$

Since the quadratic condition $x \cdot x = 1$ can only be satisfied if $x \in \{-1, 1\}$, quadratic programs can express discrete optimization problems. For example, the Max Cut problem in a graph $G = (V, E)$, where $V = \{1, \ldots, n\}$ can be written as a quadratic program in the following way:

$$\begin{array}{l} \max \sum_{ij \in E} \frac{1}{2} - \frac{1}{2} x_i \cdot x_j \\ \text{subject to} \\ x_1^2 = 1 \\ \cdots \\ x_n^2 = 1 \\ x_1, \ldots, x_n \in \mathbb{R}. \end{array} \tag{11}$$

Every quadratic program has a natural Semidefinite Programming relaxation in which we replace reals x_i with vectors v^i and we replace products $x_i \cdot x_j$ with inner products $\langle v^i, v^j \rangle$. Applying this generic transformation to the quadratic programming formulation of Max Cut, we obtain the following Semidefinite Programming formulation of Max Cut:

$$\begin{array}{l} \max \sum_{ij \in E} \frac{1}{2} - \frac{1}{2} \langle v^i, v^j \rangle \\ \text{subject to} \\ \langle v^1, v^1 \rangle = 1 \\ \cdots \\ \langle v^n, v^n \rangle = 1 \\ v^1, \ldots, v^n \in \mathbb{R}^n. \end{array} \tag{12}$$

The Max Cut relaxation (12), first studied by Delorme and Poljak [17, 16], is the one used by Goemans and Williamson.

Algorithms based on semidefinite programming provide the best known polynomial-time approximation guarantees for a number of other graph optimization problems and of constraint satisfaction problems.

4.3. **Semidefinite Programming and Unique Games.** The quality of the approximation of relaxation (12) for the Max Cut problem exactly matches the intractability results proved assuming the Unique Games Intractability Assumptions. This has been true for a number of other optimization problems.

Remarkably, Prasad Raghavendra showed in [30] that for a class of problems (which includes Max Cut as well as boolean and non-boolean constraint satisfaction problems), there is a Semidefinite Programming relaxation such that, assuming the Unique Games Intractabiltiy Conjecture, no other polynomial-time algorithm can provide a better approximation than that relaxation.

If one believes the conjecture, this means that the approximability of all such problems has been resolved, and a best-possible polynomial-time approximation algorithm has been identified for each such problem. An alternative view is that, in order to contradict the Unique Games Intractabiltiy Conjecture, it is enough to find a new algorithmic approximation techniques that work better than Semidefinite Programming for any of the problems that fall into Raghavendra's framework, or maybe to find a different Semidefinite Programming relaxation that works better than the one considered in Raghavendra's work.

4.4. **Sparsest cut, semidefinite programming, and metric embeddings.** If, at some point in the future, the Unique Games Intractability Conjecture is refuted, then some of the theorems that we have discussed will become vacuous. There are,

however, a number of unconditional results that have been discovered because of the research program that originated from the conjecture, and that would survive a refutation.

First of all, the analytic techniques developed to study reductions from Unique Games could become part of future reductions from Label Cover or from other variants of the PCP Theorem. As discussed above, reductions from Unique Games give ways of encoding values of variables of a Label Cover instance as good feasible solutions in the target optimization problems, and ways of decoding good feasible solutions in the target optimization problems as values for the variables of the Label Cover instance.

It is also worth noting that some of the analytic techniques developed within the research program of Unique Games have broader applicability. For example the impetus to prove the Invariance Theorem of Mossel, O'Donnell, and Oleszkiewicz came from its implications for conditional inapproximability results, but it settles a number of open questions in social choice theory.

Perhaps the most remarkable unconditional theorems motivated by Unique Games regard *integrality gaps* of Semidefinite Programming relaxations. The integrality gap of a relaxation of a combinatorial optimization problem is the worst-case (over all instances) ratio between the optimum of the combinatorial problem and the optimum of the relaxation. The integrality gap defines how good is the optimum of the relaxation as a numerical approximation of the true optimum, and it is usually a bottleneck to the quality of approximation algorithms that are based on the relaxation.

The integrality gap of relaxation (12) is $.8785\cdots$, the same as the hardness of approximation result proved assuming the Unique Games Intractabiltiy Conjecture. Indeed, the graph that exhibits the $.8785\cdots$ gap is related to the graph used in the reduction from Unique Games to Max Cut. (The integrality gap instance was discovered by Feige and Schechtman [19].) This is part of the larger pattern discovered by Raghavendra (cited above), who showed that, for a certain class of optimization problems, every integrality gap instance for certain Semidefinite Programming relaxations can be turned into a conditional inapproximability result assuming the Unique Games Intractability Conjecture.

The Sparsest Cut problem has a Semidefinite Programming relaxation, first studied by Goemans and Linial, whose analysis is of interest even outside of the area of approximation algorithms. A metric space (X, d) is of *negative type* if $(X, \sqrt{d})$ is also a metric space and is isometrically embeddable in Euclidean space. If every n-point metric space of negative type can be embedded into $L1$ with distortion at most $c(n)$, then the Semidefinite Programming relaxation of Goemans and Linial can be used to provide a $c(n)$-approximate algorithm for sparsest cut, where n is the number of vertices, and the integrality gap of the relaxation is at most $c(n)$. Equivalently, if there is an n-vertex instance of Sparsest Cut exhibiting an integrality gap at least $c(n)$, then there is an n-point negative-type metric space that cannot be embedded into $L1$ without incurring distortion at least $c(n)$.

Interestingly, there is a generalization of the Sparsest Cut problem, the Non-uniform Sparsest Cut problem, for which the converse is also true; that is, the integrality gap of the Goemans–Linial Semidefinite Programming relaxation of the Non-uniform Sparsest Cut problem for graphs with n vertices is $\leq c(n)$ if and only if

every n-point negative-type metric space can be embedded into $L1$ with distortion at most $c(n)$.

It had been conjectured by Goemans and Linial that the integrality gap of the semidefinite relaxations of Sparsest Cut and Non-Uniform Sparsest Cut was at most a constant. Arora, Rao, and Vazirani [8] proved in 2004 that the Sparsest Cut relaxation had integrality gap $O(\sqrt{\log n})$, and Arora, Lee, and Naor [7] proved in 2005 that Non-uniform Sparsest Cut relaxation had integrality gap $O(\sqrt{\log n} \cdot \log\log n)$, results that were considered partial progress toward the Goemans–Linial conjecture.

Later in 2005, however, Khot and Vishnoi [26] proved that the relaxation of Non-uniform Sparsest Cut has an integrality gap $(\log\log n)^{\Omega(1)}$ that goes to infinity with n. Their approach was as follows:

(1) Prove that the Non-uniform Sparsest Cut problem does not have a constant-factor approximation, assuming the Unique Games Intractability Conjecture, via a reduction from Unique Games to Non-uniform Sparsest Cut;
(2) Prove that a natural Semidefinite Programming relaxation of Unique Games has integrality gap $(\log\log n)^{\Omega(1)}$;
(3) Show that applying the reduction in (1) to the Unique Games instance in (2) produces an integrality gap instance for the Goemans–Linial Semidefinite Programming relaxation of Non-uniform Sparsest Cut.

In particular, Khot and Vishnoi exhibit an n-point negative-type metric space that requires distortion $(\log\log n)^{\Omega(1)}$ to be embedded into $L1$. This has been a rather unique approach to the construction of counterexamples in metric geometry. The lower bound was improved to $\Omega(\log\log n)$ by Krauthgamer and Rabani [27], and shortly afterward Devanur, Khot, Saket, and Vishnoi [18] showed that even the Sparsest Cut relaxation has an integrality gap $\Omega(\log\log n)$.

Cheeger, Kleiner, and Naor [13] have recently exhibited a $(\log n)^{\Omega(1)}$ integrality gap for Non-uniform Sparsest Cut, via very different techniques.

5. Algorithms for unique games

When Khot introduced the Unique Games Conjecture, he also introduced a Semidefinite Programming relaxation. Charikar, Makarychev, and Makarychev [12] provide a tight analysis of the approximation guarantee of that Semidefinite Program, showing that, given a unique game with range Σ in which a $1-\epsilon$ fraction of the equations can be satisfied, it is possible to find in polynomial time a solution that satisfies at least a $1/\Sigma^{O(\epsilon)}$ fraction of constraints.

This is about as good as can be expected, because earlier work had shown that if the Unique Games Intractability Conjecture holds, then there is no polynomial time algorithm able to satisfy a $1/\Sigma^{o_\Sigma(\epsilon)}$ fraction of constraints in a unique game with range Σ in which a $(1-\epsilon)$ fraction of equations is satisfiable. Furthermore, the analysis of Charikar, Makarychev, and Makarychev [12] is (unconditionally) known to be tight for the specific Semidefinite Programming relaxation used in their algorithm because of the integrality gap result of Khot and Vishnoi [26] discussed in the previous section.

Recently, Arora, Barak, and Steurer [6] have devised an algorithm that satisfies in time $2^{n^{\epsilon^{\Omega(1)}}}$ a constant fraction of the equations in an instance of Unique Games in which it is possible to satisfy a $1-\epsilon$ fraction of equations. Although this result is

far from refuting the Unique Games Intractability Conjecture, it casts some doubt on the Unique Games NP-hardness Conjecture. The following stronger form of the $P \neq NP$ conjecture is generally considered to be very likely: that for every NP-hard problem that is a $c > 0$ such that the problem cannot be solved with worst-case running time faster than 2^{n^c}, where n is the size of the input. This means that if the running time of the Arora–Barak–Steurer algorithm could be improved to $2^{n^{o(1)}}$ for a fixed ϵ, the Unique Games NP-hardness Conjecture would be in disagreement with the above conjecture about NP-hard problems, and it would have to be considered unlikely.

About the author

Luca Trevisan is a professor of computer science at Stanford University. He completed his Ph.D. in computer science at La Sapienza University in 1997, advised by Pierluigi Crescenzi. Before coming to Stanford, Luca taught at Columbia University and at the University of California, Berkeley. He was an invited speaker at the International Congress of Mathamaticians in 2006.

References

1. N. Alon and V.D. Milman, λ_1*, isoperimetric inequalities for graphs, and superconcentrators*, Journal of Combinatorial Theory, Series B **38** (1985), no. 1, 73–88. MR782626 (87b:05092)
2. Noga Alon, *Eigenvalues and expanders*, Combinatorica **6** (1986), no. 2, 83–96. MR875835 (88e:05077)
3. S. Arora, C. Lund, R. Motwani, M. Sudan, and M. Szegedy, *Proof verification and hardness of approximation problems*, Journal of the ACM **45** (1998), no. 3, 501–555, Preliminary version in *Proc. of FOCS '92*. MR1639346 (99d:68077b)
4. S. Arora and S. Safra, *Probabilistic checking of proofs: A new characterization of NP*, Journal of the ACM **45** (1998), no. 1, 70–122, Preliminary version in *Proc. of FOCS '92*. MR1614328 (99d:68077a)
5. Sanjeev Arora and Boaz Barak, *Computational complexity: A modern approach*, Cambridge University Press, 2009. MR2500087 (2010i:68001)
6. Sanjeev Arora, Boaz Barak, and David Steurer, *Subexponential algorithms for unique games and related problems*, Proceedings of the 51st IEEE Symposium on Foundations of Computer Science, 2010.
7. Sanjeev Arora, James Lee, and Assaf Naor, *Euclidean distortion and the sparsest cut*, Proceedings of the 37th ACM Symposium on Theory of Computing, 2005, pp. 553–562. MR2181659 (2006g:68284)
8. Sanjeev Arora, Satish Rao, and Umesh Vazirani, *Expander flows and a* $\sqrt{\log n}$*-approximation to sparsest cut*, Proceedings of the 36th ACM Symposium on Theory of Computing, 2004. MR2121604 (2005j:68081)
9. M. Bellare, O. Goldreich, and M. Sudan, *Free bits, PCP's and non-approximability – towards tight results*, SIAM Journal on Computing **27** (1998), no. 3, 804–915, Preliminary version in *Proc. of FOCS '95*. MR1612644 (2000a:68034)
10. M Bellare, S. Goldwasser, C. Lund, and A. Russell, *Efficient probabilistically checkable proofs and applications to approximation*, Proceedings of the 25th ACM Symposium on Theory of Computing, 1993, See also the errata sheet in *Proc of STOC '94*, pp. 294–304.
11. Christer Borell, *Geometric bounds on the Ornstein-Uhlenbeck velocity process*, Zeitschrift für Wahrscheinlichkeitstheorie und Verwandte Gebiete **70** (1985), 1–13. MR795785 (87k:60103)
12. Moses Charikar, Konstantin Makarychev, and Yury Makarychev, *Near-optimal algorithms for unique games*, Proceedings of the 38th ACM Symposium on Theory of Computing, 2006, pp. 205–214.
13. Jeff Cheeger, Bruce Kleiner, and Assaf Naor, *A* $(\log n)^{\Omega(1)}$ *integrality gap for the sparsest cut SDP*, arxiv:0910.2024, 2009.
14. S.A. Cook, *The complexity of theorem proving procedures*, Proceedings of the 3rd ACM Symposium on Theory of Computing, 1971, pp. 151–158.

15. Pierluigi Crescenzi, Riccardo Silvestri, and Luca Trevisan, *On weighted vs unweighted versions of combinatorial optimization problems*, Information and Compututation **167** (2001), no. 1, 10–26. MR1839896 (2003g:90058)
16. Charles Delorme and Svatopluk Poljak, *Combinatorial properties and the complexity of a max-cut approximation*, European J. of Combinatorics **14** (1993), no. 4, 313–333. MR1226579 (94e:68139)
17. ———, *Laplacian eigenvalues and the maximum cut problem*, Mathematical Programing **62** (1993), 557–574. MR1251892 (94k:90129)
18. Nikhil Devanur, Subhash Khot, Rishi Saket, and Nisheeth Vishnoi, *Integrality gaps for sparsest cut and minimum linear arrangement problems*, Proceedings of the 38th ACM Symposium on Theory of Computing, 2006, pp. 537–546. MR2277179 (2007h:68079)
19. Uriel Feige and Gideon Schechtman, *On the optimality of the random hyperplane rounding technique for MAX CUT*, Random Structures and Algorithms **20** (2002), no. 3, 403–440. MR1900615 (2003c:90086)
20. Michel X. Goemans and David P. Williamson, *Improved approximation algorithms for maximum cut and satisfiability problems using semidefinite programming*, Journal of the ACM **42** (1995), no. 6, 1115–1145, Preliminary version in *Proc. of STOC'94*. MR1412228 (97g:90108)
21. Johan Håstad, *Some optimal inapproximability results*, Journal of the ACM **48** (2001), no. 4, 798–859. MR2144931 (2006c:68066)
22. R.M. Karp, *Reducibility among combinatorial problems*, Complexity of Computer Computations (R.E. Miller and J.W. Thatcher, eds.), Plenum Press, 1972, pp. 85–103. MR0378476 (51:14644)
23. Subhash Khot, *On the power of unique 2-prover 1-round games*, Proceedings of the 34th ACM Symposium on Theory of Computing, 2002, pp. 767–775. MR2121525
24. ———, *Inapproximability of NP-complete problems, discrete Fourier analysis, and geometry*, Proceedings of the International Congress of Mathematicians, 2010.
25. Subhash Khot, Guy Kindler, Elchanan Mossel, and Ryan O'Donnell, *Optimal inapproximability results for MAX-CUT and other two-variable CSPs?*, Proceedings of the 45th IEEE Symposium on Foundations of Computer Science, 2004, pp. 146–154.
26. Subhash Khot and Nisheeth Vishnoi, *The unique games conjecture, integrality gap for cut problems and the embeddability of negative type metrics into ℓ_1*, Proceedings of the 46th IEEE Symposium on Foundations of Computer Science, 2005, pp. 53–63.
27. Robert Krauthgamer and Yuval Rabani, *Improved lower bounds for embeddings into L1*, SIAM Journal on Computing **38** (2009), no. 6, 2487–2498. MR2506299 (2010f:68238)
28. Leonid A. Levin, *Universal search problems*, Problemi Peredachi Informatsii **9** (1973), 265–266. MR0340042 (49:4799)
29. Elchanan Mossel, Ryan O'Donnell and, and Krzysztof Oleszkiewicz, *Noise stability of functions with low influences: Invariance and optimality*, Annals of Mathematics **171** (2010), no. 1, 295–341. MR2630040
30. Prasad Raghavendra, *Optimal algorithms and inapproximability results for every CSP?*, Proceedings of the 40th ACM Symposium on Theory of Computing, 2008. MR2582901
31. Ran Raz, *A parallel repetition theorem*, SIAM Journal on Computing **27** (1998), no. 3, 763–803, Preliminary version in *Proc. of STOC '95*. MR1612640 (2000c:68057)
32. V. I. Rotar, *Limit theorems for polylinear forms*, J of Multivariate Analysis **9** (1979), no. 4, 511–530. MR556909 (81m:60039)
33. Michael Sipser, *Introduction to the theory of computation*, Thomson, 2005, Second Editon.
34. L. Trevisan, G.B. Sorkin, M. Sudan, and D.P. Williamson, *Gadgets, approximation, and linear programming*, SIAM Journal on Computing **29** (2000), no. 6, 2074–2097. MR1756405 (2001j:68046)
35. Vijay Vazirani, *Approximation algorithms*, Springer, 2001. MR1851303 (2002h:68001)

Computer Science Department, Stanford University, 353 Serra Mall Stanford, California 94305-9025

E-mail address: trevisan@stanford.edu