Unique Games Conjecture

Konstantinos X. Koiliaris

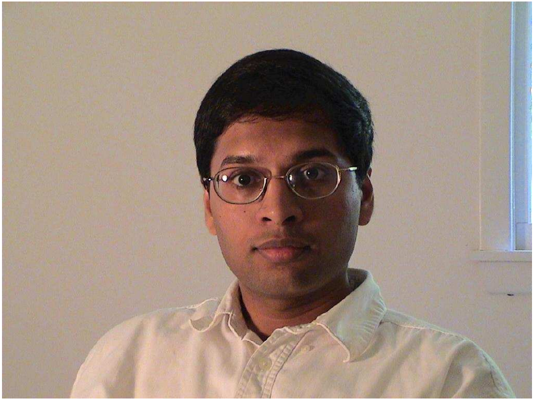
$\mu \prod \lambda \forall$

February 7, 2013

Table of Contents

The Culprit

The Culprit



Subhash Khot

The Label Cover Problem

In the Label Cover (LC) problem we are given

The Label Cover Problem

In the Label Cover (LC) problem we are given

- A graph $G = (V, E)$;

The Label Cover Problem

In the Label Cover (LC) problem we are given

- A graph $G = (V, E)$;
- A set of labels L; and

The Label Cover Problem

In the Label Cover (LC) problem we are given

- A graph $G = (V, E)$;
- A set of labels L; and
- For each edge $e = (u, v) \in E$, a set $R_e \subseteq L \times L$ consisting of a set of "permissible" values for the pair (u, v).

The Label Cover Problem

In the Label Cover (LC) problem we are given

- A graph $G = (V, E)$;
- A set of labels L; and
- For each edge $e = (u, v) \in E$, a set $R_e \subseteq L \times L$ consisting of a set of "permissible" values for the pair (u, v).

The goal is to assign a label $l_v \in L$ to each vertex $v \in V$ in such a way to satisfy the maximum number of edges.

The Label Cover Problem

In the Label Cover (LC) problem we are given

- A graph $G = (V, E)$;
- A set of labels L; and
- For each edge $e = (u, v) \in E$, a set $R_e \subseteq L \times L$ consisting of a set of "permissible" values for the pair (u, v).

The goal is to assign a label $l_v \in L$ to each vertex $v \in V$ in such a way to satisfy the maximum number of edges.

We say that an edge $e = (u, v)$ is satisfied if $(l_u, l_v) \in R_e$.

The Label Cover Problem

In the Label Cover (LC) problem we are given

- A graph $G = (V, E)$;
- A set of labels L; and
- For each edge $e = (u, v) \in E$, a set $R_e \subseteq L \times L$ consisting of a set of "permissible" values for the pair (u, v).

The goal is to assign a label $l_v \in L$ to each vertex $v \in V$ in such a way to satisfy the maximum number of edges.

We say that an edge $e = (u, v)$ is satisfied if $(l_u, l_v) \in R_e$.

We denote by $\mathsf{VAL}(\mathsf{LC}) \in [0, 1]$ the maximum possible fraction of satisfied edges by any labeling.

Label Cover Example #1

We can view 3-Coloring as a Label Cover problem.

Label Cover Example #1

We can view 3-Coloring as a Label Cover problem.

- Use the same graph,

Label Cover Example #1

We can view 3-Coloring as a Label Cover problem.

- Use the same graph,
- $L = \{1, 2, 3\}$,

Label Cover Example #1

We can view 3-Coloring as a Label Cover problem.

- Use the same graph,
- $L = \{1, 2, 3\}$,
- $R_e = \{(1, 2), (1, 3), (2, 1), (2, 3), (3, 1), (3, 2)\}$ for all edges,

Label Cover Example #1

We can view 3-Coloring as a Label Cover problem.

- Use the same graph,
- $L = \{1, 2, 3\}$,
- $R_e = \{(1,2), (1,3), (2,1), (2,3), (3,1), (3,2)\}$ for all edges,
- $\mathsf{VAL}(\mathsf{LC}) = 1 \Leftrightarrow G$ is 3-colorable.

Label Cover Example #1

We can view 3-Coloring as a Label Cover problem.

- Use the same graph,
- $L = \{1, 2, 3\}$,
- $R_e = \{(1,2), (1,3), (2,1), (2,3), (3,1), (3,2)\}$ for all edges,
- $\mathsf{VAL}(\mathsf{LC}) = 1 \Leftrightarrow G$ is 3-colorable.

Thus, it is NP-hard to determine if we can satisfy all edges. Ok, so Label Cover is a hard problem!

Label Cover Example #2

We can also view 3SAT as a Label Cover problem, as follows:

Label Cover Example #2

We can also view 3SAT as a Label Cover problem, as follows:

- We assign our vertex set to be $V = \{\text{variables}\} \cup \{\text{clauses}\}$,

Label Cover Example #2

We can also view 3SAT as a Label Cover problem, as follows:

- We assign our vertex set to be $V = \{\text{variables}\} \cup \{\text{clauses}\}$,
- Create an edge between clause ϕ and variable x, if x occurs in ϕ,

Label Cover Example #2

We can also view 3SAT as a Label Cover problem, as follows:

- We assign our vertex set to be $V = \{\text{variables}\} \cup \{\text{clauses}\}$,
- Create an edge between clause ϕ and variable x, if x occurs in ϕ,
- $L = \{000, 001, 010, 011, 100, 101, 110, 111, 0, 1\}$,

Label Cover Example #2

We can also view 3SAT as a Label Cover problem, as follows:

- We assign our vertex set to be $V = \{\text{variables}\} \cup \{\text{clauses}\}$,
- Create an edge between clause ϕ and variable x, if x occurs in ϕ,
- $L = \{000, 001, 010, 011, 100, 101, 110, 111, 0, 1\}$,
- We say that an edge $e = (\phi, x)$ is satisfied when:

Label Cover Example #2

We can also view 3SAT as a Label Cover problem, as follows:

- We assign our vertex set to be $V = \{\text{variables}\} \cup \{\text{clauses}\}$,
- Create an edge between clause ϕ and variable x, if x occurs in ϕ,
- $L = \{000, 001, 010, 011, 100, 101, 110, 111, 0, 1\}$,
- We say that an edge $e = (\phi, x)$ is satisfied when:
  - The label for ϕ is one of the seven satisfying assignments for the three variables in ϕ, and

Label Cover Example #2

We can also view 3SAT as a Label Cover problem, as follows:

- We assign our vertex set to be $V = \{\text{variables}\} \cup \{\text{clauses}\}$,
- Create an edge between clause ϕ and variable x, if x occurs in ϕ,
- $L = \{000, 001, 010, 011, 100, 101, 110, 111, 0, 1\}$,
- We say that an edge $e = (\phi, x)$ is satisfied when:
  - The label for ϕ is one of the seven satisfying assignments for the three variables in ϕ, and
  - The labels for x andϕ give the same value to x.

2-Prover Games

Consider a 2-Prover Game:

2-Prover Games

Consider a 2-Prover Game:

- Two provers try to convince us (the verifier) that some 3-CNF formula is satisfiable,

2-Prover Games

Consider a 2-Prover Game:

- Two provers try to convince us (the verifier) that some 3-CNF formula is satisfiable,
- We send a question to each of the provers, which they have to answer independently.

2-Prover Games

Consider a 2-Prover Game:

- Two provers try to convince us (the verifier) that some 3-CNF formula is satisfiable,
- We send a question to each of the provers, which they have to answer independently.

We can view this as a Label Cover problem:

2-Prover Games

Consider a 2-Prover Game:

- Two provers try to convince us (the verifier) that some 3-CNF formula is satisfiable,
- We send a question to each of the provers, which they have to answer independently.

We can view this as a Label Cover problem:

- $V = \{\text{the set of possible questions}\}$,

2-Prover Games

Consider a 2-Prover Game:

- Two provers try to convince us (the verifier) that some 3-CNF formula is satisfiable,
- We send a question to each of the provers, which they have to answer independently.

We can view this as a Label Cover problem:

- $V = \{\text{the set of possible questions}\}$,
- $L = \{\text{the set of possible answers}\}$,

2-Prover Games

Consider a 2-Prover Game:

- Two provers try to convince us (the verifier) that some 3-CNF formula is satisfiable,
- We send a question to each of the provers, which they have to answer independently.

We can view this as a Label Cover problem:

- $V = \{\text{the set of possible questions}\}$,
- $L = \{\text{the set of possible answers}\}$,
- An edge $e = (u, v)$ is satisfied by a pair of answers if that pair of answers make the verifier accept.

2-Prover Games

Consider a 2-Prover Game:

- Two provers try to convince us (the verifier) that some 3-CNF formula is satisfiable,
- We send a question to each of the provers, which they have to answer independently.

We can view this as a Label Cover problem:

- $V = \{\text{the set of possible questions}\}$,
- $L = \{\text{the set of possible answers}\}$,
- An edge $e = (u, v)$ is satisfied by a pair of answers if that pair of answers make the verifier accept.

Notice now that finding a good strategy for the provers is equivalent to solving the label cover problem!

Approximating Label Cover

It follows from *PCP theorem* and *Parallel Repetition theorem* that:

Approximating Label Cover

It follows from *PCP theorem* and *Parallel Repetition theorem* that:

Theorem

For every $n > 0$ there exists an L such that it is NP*-hard to distinguish* Label Cover *instances,* LC*, with* $\mathsf{VAL}(\mathsf{LC}) = 1$ *from those with* $\mathsf{VAL}(\mathsf{LC}) \leq n$ *for instances* LC *with the provided label set L.*

Approximating Label Cover

It follows from *PCP theorem* and *Parallel Repetition theorem* that:

Theorem

For every $n > 0$ there exists an L such that it is NP*-hard to distinguish* *Label Cover instances,* LC*, with* $\mathsf{VAL}(\mathsf{LC}) = 1$ *from those with* $\mathsf{VAL}(\mathsf{LC}) \leq n$ *for instances* LC *with the provided label set L.*

Ok, hold that thought for a second!

Unique Label Cover

Unique Label Cover is the special case of Label Cover when the relation for each edge is a bijection.

Unique Label Cover

Unique Label Cover is the special case of Label Cover when the relation for each edge is a bijection.

- In other words, for each edge $e = (u, v)$ and choice of label for u there is exactly one choice of label for v that satisfies the edge e (and vice versa),

Unique Label Cover

Unique Label Cover is the special case of Label Cover when the relation for each edge is a bijection.

- In other words, for each edge $e = (u, v)$ and choice of label for u there is exactly one choice of label for v that satisfies the edge e (and vice versa),
- Equivalently, in the 2-Prover Game, given the answer from one of the provers, there is exactly one answer from the other prover that will make the verifier accept

Unique Label Cover

Unique Label Cover is the special case of Label Cover when the relation for each edge is a bijection.

- In other words, for each edge $e = (u, v)$ and choice of label for u there is exactly one choice of label for v that satisfies the edge e (and vice versa),
- Equivalently, in the 2-Prover Game, given the answer from one of the provers, there is exactly one answer from the other prover that will make the verifier accept

So, naturally, the question arises:

Unique Label Cover

Unique Label Cover is the special case of Label Cover when the relation for each edge is a bijection.

- In other words, for each edge $e = (u, v)$ and choice of label for u there is exactly one choice of label for v that satisfies the edge e (and vice versa),
- Equivalently, in the 2-Prover Game, given the answer from one of the provers, there is exactly one answer from the other prover that will make the verifier accept

So, naturally, the question arises:

Unique Label Cover Hardness

How hard is Unique Label Cover?

Unique Games Conjecture (UGC)

Conjecture (Khot 2002)

For any $n > 0$ there exists an L such that it is NP*-hard to distinguish* *Unique Label Cover* *instances with* $\mathsf{VAL}(\mathsf{ULC}) > 1 - n$ *from those with* $\mathsf{VAL}(\mathsf{ULC}) \leq n$ *for instances* ULC *with the provided label set L.*

Motivation

There are two reasons for which UGC is particularly intriguing.

Motivation

There are two reasons for which UGC is particularly intriguing.

First, it is a well-balanced question.

Motivation

There are two reasons for which UGC is particularly intriguing.

First, it is a well-balanced question.

Despite continuous efforts to prove or disprove it, there is still no consensus regarding its validity.

Motivation

There are two reasons for which UGC is particularly intriguing.

First, it is a well-balanced question.

Despite continuous efforts to prove or disprove it, there is still no consensus regarding its validity.

This seems to indicate that UGC is more likely to be resolved in the near future in contrast to the P-NP problem for example, for which it is widely believed that P$\neq$NP but current techniques have not been able to prove it.

Motivation

Second, the truth of UGC implies that the currently best known approximation algorithms for many important computational problems have optimal approximation ratios.

Motivation

Second, the truth of UGC implies that the currently best known approximation algorithms for many important computational problems have optimal approximation ratios.

Since its origin, UGC has been successfully used to prove, often optimal, hardness of approximation results for several important NP-hard problems, such as

- Min-2Sat-Deletion (Khot 2002);

Motivation

Second, the truth of UGC implies that the currently best known approximation algorithms for many important computational problems have optimal approximation ratios.

Since its origin, UGC has been successfully used to prove, often optimal, hardness of approximation results for several important NP-hard problems, such as

- Min-2Sat-Deletion (Khot 2002);
- Vertex Cover (Khot & Regev 2003);

Motivation

Second, the truth of UGC implies that the currently best known approximation algorithms for many important computational problems have optimal approximation ratios.

Since its origin, UGC has been successfully used to prove, often optimal, hardness of approximation results for several important NP-hard problems, such as

- Min-2Sat-Deletion (Khot 2002);
- Vertex Cover (Khot & Regev 2003);
- Maximum Cut (Khot *et al.* 2004);

Motivation

Second, the truth of UGC implies that the currently best known approximation algorithms for many important computational problems have optimal approximation ratios.

Since its origin, UGC has been successfully used to prove, often optimal, hardness of approximation results for several important NP-hard problems, such as

- Min-2Sat-Deletion (Khot 2002);
- Vertex Cover (Khot & Regev 2003);
- Maximum Cut (Khot *et al.* 2004);
- ...

Motivation

Second, the truth of UGC implies that the currently best known approximation algorithms for many important computational problems have optimal approximation ratios.

Since its origin, UGC has been successfully used to prove, often optimal, hardness of approximation results for several important NP-hard problems, such as

- Min-2Sat-Deletion (Khot 2002);
- Vertex Cover (Khot & Regev 2003);
- Maximum Cut (Khot *et al.* 2004);
- ...

In addition, in recent years, UGC has also proved to be intimately connected to the limitations of Semidefinite Programming (SDP). In particular, if UGC is true, then for every Constraint Satisfaction Problem (CSP) the best approximation ratio is given by a certain simple SDP.

Table of Contents

Inapproximability Preliminaries

Let $\mathcal{I}$ denote an NP-complete problem.

Inapproximability Preliminaries

Let $\mathcal{I}$ denote an NP-complete problem.

For an instance I of the problem with input size n, let $\mathsf{OPT}(I)$ denote the value of the optimal solution and for a specific polynomial time approximation algorithm, let $\mathsf{ALG}(I)$ denote the value of the solution it finds.

Inapproximability Preliminaries

Let $\mathcal{I}$ denote an NP-complete problem.

For an instance I of the problem with input size n, let $\mathsf{OPT}(I)$ denote the value of the optimal solution and for a specific polynomial time approximation algorithm, let $\mathsf{ALG}(I)$ denote the value of the solution it finds.

We say the problem $\mathcal{I}$ is UG-hard if Unique Label Cover can be efficiently reduced to $\mathcal{I}$.

Inapproximability Preliminaries

Let $\mathcal{I}$ denote an NP-complete problem.

For an instance I of the problem with input size n, let $\mathsf{OPT}(I)$ denote the value of the optimal solution and for a specific polynomial time approximation algorithm, let $\mathsf{ALG}(I)$ denote the value of the solution it finds.

We say the problem $\mathcal{I}$ is UG-hard if Unique Label Cover can be efficiently reduced to $\mathcal{I}$.

Notice that under the UGC, UG-hard $\Rightarrow$ NP-hard!

Minimum Vertex Cover (Min-VC)

As we know Min-VC is trivial to approximate within a factor of 2 (*while there are edges which are not covered, include both vertices of one such edge in the cover*).

Minimum Vertex Cover (Min-VC)

As we know Min-VC is trivial to approximate within a factor of 2 (*while there are edges which are not covered, include both vertices of one such edge in the cover*).

Dinur and Safra proved in 2002 that it is NP-hard to approximate Min-VC within 1.36.

Minimum Vertex Cover (Min-VC)

As we know Min-VC is trivial to approximate within a factor of 2 (*while there are edges which are not covered, include both vertices of one such edge in the cover*).

Dinur and Safra proved in 2002 that it is NP-hard to approximate Min-VC within 1.36.

By using the UGC, Khot and Regev proved in 2003 that it is UG-hard to approximate Min-VC within $2 - \varepsilon$ for any $\varepsilon > 0$

Maximum Independent Set (Max-IS)

We know that Max-IS is easy to approximate within $\frac{1}{\Delta+1}$, where $\Delta = \max_{v \in V} deg(v)$ (...).

Maximum Independent Set (Max-IS)

We know that Max-IS is easy to approximate within $\frac{1}{\Delta+1}$, where $\Delta = \max_{v \in V} deg(v)$ (...).

Trevisan proved in 2001 that it is NP-hard to approximate Max-IS within Min-VC within $\frac{2^{c\sqrt{\Delta}}}{\Delta}$, for $c > 0$.

Maximum Independent Set (Max-IS)

We know that Max-IS is easy to approximate within $\frac{1}{\Delta+1}$, where $\Delta = \max_{v \in V} deg(v)$ (...).

Trevisan proved in 2001 that it is NP-hard to approximate Max-IS within Min-VC within $\frac{2^{c\sqrt{\Delta}}}{\Delta}$, for $c > 0$.

By using the UGC, Samorodnitsky and Trevisan proved in 2005 that it is UG-hard to approximate Max-IS within $\frac{(\log \Delta)^c}{\Delta}$, for some constant $c > 0$.

Almost 3-coloring

Given a graph $G = (V, E)$, Almost 3-coloring is defined as the problem of removing enough vertices to get a 3-colorable graph.

Almost 3-coloring

Given a graph $G = (V, E)$, Almost 3-coloring is defined as the problem of removing enough vertices to get a 3-colorable graph.

We will call this number $R(G)$ and our goal will be to minimize it.

Almost 3-coloring

Given a graph $G = (V, E)$, Almost 3-coloring is defined as the problem of removing enough vertices to get a 3-colorable graph.

We will call this number $R(G)$ and our goal will be to minimize it.

Dinur, Mossel, and Regev proved in 2005 that it is UG-hard to distinguish between $R(G) < \varepsilon|V|$ and $R(G) \geq (1-\varepsilon)|V|$ for any $\varepsilon > 0$.

Maximum Cut (Max-Cut)

We know that Max-Cut is easy to approximate within a factor of $\frac{1}{2}$ (*start with an arbitrary partition of the vertices of the graph and repeatedly move one vertex at a time from one side of the partition to the other, improving the solution at each step, until no more improvements of this type can be made*).

Maximum Cut (Max-Cut)

We know that Max-Cut is easy to approximate within a factor of $\frac{1}{2}$ (*start with an arbitrary partition of the vertices of the graph and repeatedly move one vertex at a time from one side of the partition to the other, improving the solution at each step, until no more improvements of this type can be made*).

Goemans and Williamson proved in 1995 that Max-Cut can be approximated to within a factor $\alpha_{GW} \approx 0.8785$

Maximum Cut (Max-Cut)

We know that Max-Cut is easy to approximate within a factor of $\frac{1}{2}$ (*start with an arbitrary partition of the vertices of the graph and repeatedly move one vertex at a time from one side of the partition to the other, improving the solution at each step, until no more improvements of this type can be made*).

Goemans and Williamson proved in 1995 that Max-Cut can be approximated to within a factor $\alpha_{GW} \approx 0.8785$

Håstad proved in 2001 that it is NP-hard to approximate Max-Cut within $\frac{16}{17} + \varepsilon \approx 0.9418$

Maximum Cut (Max-Cut)

We know that Max-Cut is easy to approximate within a factor of $\frac{1}{2}$ (*start with an arbitrary partition of the vertices of the graph and repeatedly move one vertex at a time from one side of the partition to the other, improving the solution at each step, until no more improvements of this type can be made*).

Goemans and Williamson proved in 1995 that Max-Cut can be approximated to within a factor $\alpha_{GW} \approx 0.8785$

Håstad proved in 2001 that it is NP-hard to approximate Max-Cut within $\frac{16}{17} + \varepsilon \approx 0.9418$

Khot, Kindler, Mossel and O'Donnell proved in 2004 that it is UG-hard to approximate Max-Cut within $\alpha_{GW} + \varepsilon$, for any $\varepsilon > 0$

Maximum 2SAT (Max-2SAT)

Very similar to Max-2SAT, Max-2SAT is the problem of finding an assignment which satisfies as many clauses as possible in a given 2-CNF formula.

Maximum 2SAT (Max-2SAT)

Very similar to Max-2SAT, Max-2SAT is the problem of finding an assignment which satisfies as many clauses as possible in a given 2-CNF formula.

We know that Max-2SAT is easy to approximate within a factor 3/4.

Maximum 2SAT (Max-2SAT)

Very similar to Max-2SAT, Max-2SAT is the problem of finding an assignment which satisfies as many clauses as possible in a given 2-CNF formula.

We know that Max-2SAT is easy to approximate within a factor 3/4.

Håstad proved in 2001 that it is NP-hard to approximate Max-2SAT within $\frac{21}{22} + \varepsilon \approx 0.9546$.

Maximum 2SAT (Max-2SAT)

Very similar to Max-2SAT, Max-2SAT is the problem of finding an assignment which satisfies as many clauses as possible in a given 2-CNF formula.

We know that Max-2SAT is easy to approximate within a factor 3/4.

Håstad proved in 2001 that it is NP-hard to approximate Max-2SAT within $\frac{21}{22} + \varepsilon \approx 0.9546$.

Lewin, Livnat and Zwick proved in 2002 that Max-2SAT can apparently be approximated to within a factor $\alpha_{LLZ} \approx 0.9401$.

Maximum 2SAT (Max-2SAT)

Very similar to Max-2SAT, Max-2SAT is the problem of finding an assignment which satisfies as many clauses as possible in a given 2-CNF formula.

We know that Max-2SAT is easy to approximate within a factor 3/4.

Håstad proved in 2001 that it is NP-hard to approximate Max-2SAT within $\frac{21}{22} + \varepsilon \approx 0.9546$.

Lewin, Livnat and Zwick proved in 2002 that Max-2SAT can apparently be approximated to within a factor $\alpha_{LLZ} \approx 0.9401$.

Austrin proved in 2006 that it is UG-hard to approximate Max-2SAT within $\alpha_{LLZ} + \varepsilon$, for any $\varepsilon > 0$.

..More Results

Problem	Best Approx. Known	Best Inapprox. Under UGC	Best Inapprox. Known
Max-kCSP	$O(2^k/k)$	$\Omega(2^k/k)$	$2^{k-O(\sqrt{k})}$
Max Acyclic Subgraph	2	$2-\varepsilon$	$\frac{66}{65}-\varepsilon$
Feedback Arc Set	$O(\log N)$	$\omega(1)$	APX-hard
Non-uni. Sparsest Cut	$O(\log N)$	$\omega(1)$	APX-hard
Min-2SAT-Deletion, Min-Uncut	$O(\sqrt{\log N})$	$\omega(1)$	APX-hard
Coloring 3-colorable Graphs	$N{\cdot}2111$	$\omega(1)$	5
Multiway Cut integr. gap $a \leq 1.344$	α	$\alpha-\varepsilon$	APX-hard

Unique Label Cover Inapproximability

As a final note, we state this result by Feige and Reichman in 2004, without its proof.

Theorem (Feige & Reichman 2004)

For any $n > 0$ there exists a $\gamma \in (0, 1)$ and an L such that it is NP*-hard to distinguish between* Unique Label Cover *instances with* $\mathsf{VAL}(\mathsf{ULC}) \geq \gamma$ *and*$\mathsf{VAL}(\mathsf{ULC}) < n\gamma$ *for instances* ULC *with the provided label set L.*

Table of Contents

Final Words

If the UGC proven correct, we would have all the "right" inapproximability results in a unifying way, nicely fit together.

Final Words

If the UGC proven correct, we would have all the "right" inapproximability results in a unifying way, nicely fit together.

On the other hand, proving the conjecture incorrect would be an algorithmic breakthrough, likely going beyond the SDP barrier, since the UGC seems to exactly capture the limitations of SDPs.

Final Words

If the UGC proven correct, we would have all the "right" inapproximability results in a unifying way, nicely fit together.

On the other hand, proving the conjecture incorrect would be an algorithmic breakthrough, likely going beyond the SDP barrier, since the UGC seems to exactly capture the limitations of SDPs.

Irrespective of its truth, the conjecture will probably lead to even more techniques and unconditional results.

References

- Subhash Khot: On the Unique Games Conjecture (Invited Survey). IEEE Conference on Computational Complexity 2010: 99-121.
- S. Khot, P. Popat, and R. Saket. Approximate Lasserre integrality gap for unique games. Manuscript, 2010.
- U. Feige and D. Reichman. On systems of linear equations with two variables per equation. In APPROX-RANDOM, pages 117-127, 2004.
- C. Papadimitriou and M. Yannakakis. Optimization, approximation, and complexity classes. Journal of Computer and Systems Sciences, 43:425-440, 1991.
- N. Linial. Finite metric spaces-combinatorics, geometry and algorithms. In Proc. International Congress of Mathematicians, volume 3, pages 573-586, 2002.
- M. X. Goemans. Semidefinite programming in combinatorial optimization. Math. Program., 79:143-161, 1997.

Thank you!