From Weak to Strong LP Gaps for all CSPs

Mrinalkanti Ghosh
joint work with: Madhur Tulsiani
MAX \( k \)-CSP

- \( n \) variables
- \( m \) constraints
MAX k-CSP

- $n$ variables taking boolean values.
- $m$ constraints: each is a k-ary boolean predicate.
- Satisfy as many as possible.
MAX k-CSP

- $n$ variables taking boolean values.
- $m$ constraints: each is a $k$-ary boolean predicate.
- Satisfy as many as possible.

Max-3-SAT

$x_1 \lor x_{22} \lor \overline{x}_{19}$

$x_3 \lor \overline{x}_9 \lor x_{23}$

$x_5 \lor \overline{x}_7 \lor \overline{x}_9$

:\
MAX k-CSP

- \( n \) variables taking boolean values.
- \( m \) constraints: each is a k-ary boolean predicate.
- Satisfy as many as possible.

Max-3-SAT

\[
\begin{align*}
  x_1 & \lor x_22 \lor \overline{x}_{19} \\
  x_3 & \lor \overline{x}_9 \lor x_{23} \\
  x_5 & \lor \overline{x}_7 \lor \overline{x}_9 \\
  \vdots & \vdots
\end{align*}
\]

Max-Cut

\[
\begin{align*}
  x_1 & \neq x_2 \\
  x_2 & \neq x_5 \\
  x_3 & \neq x_4 \\
  \vdots & \vdots
\end{align*}
\]
MAX $k$-CSP

- $n$ variables taking boolean values.
- $m$ constraints: each is a $k$-ary boolean predicate.
- Satisfy as many as possible.

### Max-3-SAT

\[
x_1 \lor x_{22} \lor \overline{x}_{19} \\
x_3 \lor \overline{x}_9 \lor x_{23} \\
x_5 \lor \overline{x}_7 \lor \overline{x}_9 \\
\vdots
\]

### Max-Cut

\[
x_1 \neq x_2 \\
x_2 \neq x_5 \\
x_3 \neq x_4 \\
\vdots
\]

Approximation Problem: Approximate the fraction of constraints simultaneously satisfiable.
MAX $k$-CSP

- $n$ variables taking values in some finite domains.
- $m$ constraints: each is a non-negative $k$-ary function.
- Satisfy as many as possible.

Max-3-SAT

\[ \begin{align*}
x_1 \lor x_{22} \lor \overline{x}_{19} \\
x_3 \lor \overline{x}_9 \lor x_{23} \\
x_5 \lor \overline{x}_7 \lor \overline{x}_9 \\
\vdots
\end{align*} \]

Max-Cut

\[ \begin{align*}
x_1 \neq x_2 \\
x_2 \neq x_5 \\
x_3 \neq x_4 \\
\vdots
\end{align*} \]

Approximation Problem: Approximate the fraction of constraints simultaneously satisfiable.
CSPs and Relaxations

MAX k-CSP (f): for $i$-th constraint, let $S_{C_i} := (x_{i_1}, \cdots, x_{i_k})$. Then:

$$C_i \equiv f(x_{i_1} + b_{i_1}, \cdots, x_{i_k} + b_{i_k}) \equiv \sum_{\alpha \in \{0,1\}^{S_{C_i}}} f(\alpha + b_{C_i}) \cdot x(S_{C_i}, \alpha),$$

with $x(S_{C_i}, \alpha) =$ indicator of assignment of $\alpha$ to $S_{C_i}$. 

CSPs and Relaxations

MAX k-CSP (f): for $i$-th constraint, let $S_{Ci} := (x_{i_1}, \cdots, x_{i_k})$. Then:

$$C_i \equiv f(x_{i_1} + b_{i_1}, \cdots, x_{i_k} + b_{i_k}) \equiv \sum_{\alpha \in \{0,1\}^{S_{Ci}}} f(\alpha + b_{Ci}) \cdot x(S_{Ci},\alpha),$$

with $x(S_{Ci},\alpha) =$ indicator of assignment of $\alpha$ to $S_{Ci}$.

$$\sum_{\alpha \in \{0,1\}^{S_{Ci}}} x(S_{Ci},\alpha) = x(i,b) \quad \forall C \in \Phi, i \in S_{C}, \quad b \in \{0,1\}$$

$$\sum_{b \in \{0,1\}} x(i,b) = 1 \quad \forall i \in [n]$$

$$x(S,\alpha) \geq 0$$
CSPs and Relaxations

MAX k-CSP (f): for $i$-th constraint, let $S_{C_i} := (x_{i_1}, \cdots, x_{i_k})$. Then:

$$C_i \equiv f(x_{i_1} + b_{i_1}, \cdots, x_{i_k} + b_{i_k}) \equiv \sum_{\alpha \in \{0,1\}^{S_{C_i}}} f(\alpha + b_{C_i}) \cdot x(S_{C_i}, \alpha),$$

with $x(S_{C_i}, \alpha) =$ indicator of assignment of $\alpha$ to $S_{C_i}$.

maximize $\mathbb{E}_{C \in \Phi} \left[ \sum_{\alpha \in \{0,1\}^{S_C}} f(\alpha + b_C) \cdot x(S_C, \alpha) \right]$}

$$\sum_{\alpha \in \{0,1\}^{S_C}} x(S_C, \alpha) = x(i,b) \quad \forall C \in \Phi, i \in S_C, b \in \{0,1\}$$

$$\sum_{b \in \{0,1\}} x(i,b) = 1 \quad \forall i \in [n]$$

$$x(S, \alpha) \geq 0$$
CSPs and Relaxations

MAX k-CSP (f): for $i$-th constraint, let $S_{C_i} := (x_{i_1}, \cdots, x_{i_k})$. Then:

$$C_i \equiv f(x_{i_1} + b_{i_1}, \cdots, x_{i_k} + b_{i_k}) \equiv \sum_{\alpha \in \{0,1\}^{S_{C_i}}} f(\alpha + b_{C_i}) \cdot x(s_{C_i}, \alpha),$$

with $x(s_{C_i}, \alpha) = \text{indicator of assignment of } \alpha \text{ to } S_{C_i}$.

maximize $\mathbb{E}_{C \in \Phi} \left[ \sum_{\alpha \in \{0,1\}^{S_C}} f(\alpha + b) \cdot x(s, \alpha) \right]$

$$\sum_{\alpha \in \{0,1\}^{S_C}} x(s, \alpha) = x(i, b) \quad \forall C \in \Phi, i \in S_C, \quad b \in \{0,1\}$$

$$\sum_{b \in \{0,1\}} x(i, b) = 1 \quad \forall i \in [n]$$

$$x(s, \alpha) \geq 0$$

#constraints $= \Theta (m \cdot 2^k)$
- **Extended Formulation**: Defined by a feasible polytope $P$, and a way of encoding instances $\Phi$ as a (linear) objective function $w_\Phi$. 

- Feasible point in $\text{SA}(t)$: Family $\{D_S\} | |S| \leq t$ of consistent distribution with $D_S$ a distribution on $\{0, 1\}^S$. 

- Similarly, for Basic LP solution.
Extended Formulation and Sherali-Adams Relaxation

- **Extended Formulation:** Defined by a feasible polytope $P$, and a way of encoding instances $\Phi$ as a (linear) objective function $w_\Phi$.

- Optimize objective $\langle w_\Phi, x \rangle$ (depending on $\Phi$) over $P$. 

Extended Formulation and Sherali-Adams Relaxation

- **Extended Formulation**: Defined by a feasible polytope $P$, and a way of encoding instances $\Phi$ as a (linear) objective function $w_\Phi$.

- Optimize objective $\langle w_\Phi, x \rangle$ (depending on $\Phi$) over $P$.

- Introduce additional variables $y$. Optimize over polytope $P = \{x \mid \exists y \text{ } Ex + Fy = g, y \geq 0\}$. Size equals $\#\text{variables} + \#\text{constraints}$.
Extended Formulation and Sherali-Adams Relaxation

- **Extended Formulation**: Defined by a feasible polytope $P$, and a way of encoding instances $\Phi$ as a (linear) objective function $w_\Phi$.

- **Sherali-Adams**: A Sherali-Adams of level $t$ is an Extended Formulation with
  \[ \#\text{variables} = \binom{n}{t} \cdot 2^t. \]
Extended Formulation and Sherali-Adams Relaxation

- **Extended Formulation**: Defined by a feasible polytope $P$, and a way of encoding instances $\Phi$ as a (linear) objective function $w_\Phi$.

- **Sherali-Adams**: A Sherali-Adams of level $t$ is an Extended Formulation with
  \#variables $= \binom{n}{t} \cdot 2^t$.

- Variables: $x(S, \alpha)$, $|S| \leq t$, $\alpha \in \{0, 1\}^S$. 

Image from [Fiorini-Rothvoss-Tiwari-11]
Extended Formulation and Sherali-Adams Relaxation

**EF:**

- **Extended Formulation:** Defined by a feasible polytope $P$, and a way of encoding instances $\Phi$ as a (linear) objective function $w_\Phi$.

**SA:**

- **Sherali-Adams:** A Sherali-Adams of level $t$ is an Extended Formulation with
  \[ \# \text{variables} = \binom{n}{t} \cdot 2^t. \]
- Feasible point in $SA(t)$: Family $\{D_S\}_{|S|\leq t}$ of consistent distribution with $D_S$ a distribution on $\{0, 1\}^S$. 
- **Extended Formulation**: Defined by a feasible polytope $P$, and a way of encoding instances $\Phi$ as a (linear) objective function $w_\Phi$.

- **Sherali-Adams**: A Sherali-Adams of level $t$ is an Extended Formulation with
  
  $\#\text{variables} = \binom{n}{t} \cdot 2^t$.

- Feasible point in $\text{SA}(t)$: Family $\{\mathcal{D}_S\}_{|S|\leq t}$ of consistent distribution with $\mathcal{D}_S$ a distribution on $\{0, 1\}^S$.

- Similarly, for Basic LP solution.
Result

Basic LP

Sherali-Adams

LP Extended Formulation

[CLRS13 KMR17]
Main Theorem: For all CSPs, if Basic LP has integrality gap of \((c, s)\) then for all \(\varepsilon > 0\), there exist large enough instance(s) with integrality gap of \((c - \varepsilon, s + \varepsilon)\) for \(SA(\tilde{O}_\varepsilon(\log n))\).
Result

With [Kothari-Meka-Raghavendra-17]: For all CSPs, if Basic LP has \((c, s)\) gap, then so does any LP Extended Formulation of size \(n^{\tilde{O}(\log n)}\).

Ignoring \(\epsilon\) losses.
Hard Instance

- For each variable in $\Phi_0$, create bucket with large number of variables.
- Independently, sample each constraint as:
  - Sample constraint $C$ from $\Phi_0$.
  - For each variable $x$ in $S_C$, choose $y_x \in B_x$, u.a.r.
  - Put the constraint $C$ on the variables $\{y_x\}_{x \in S_C}$.

W.h.p., the instance hypergraph generated has $o(n)$ cycles of length at most $\eta \log n$ for $\eta > 0$.

Remove one constraint from every small cycle and get an instance of girth $\eta \log n$. 
Use the hard instance $\Phi_0$ of the basic relaxation as template to build the new hard instance on $n$ variables and $m = \Delta \cdot n$ constraints.
Hard Instance

#variables = n and #constraints = m = Δ · n.
Hard Instance

#variables = n and #constraints = m = Δ · n.

- For each variable in \( \Phi_0 \), create bucket with large number of variables.
Hard Instance

\#variables = n and \#constraints = m = \Delta \cdot n.

- For each variable in \( \Phi_0 \), create bucket with large number of variables.
- Independently, sample each constraint as:

\text{Sample constraint } C \text{ from } \Phi_0.

For each variable \( x \) in \( S_C \), choose \( y_x \in B_x \), u.a.r.

Put the constraint \( C \) on the variables \( \{y_x\} \).

W.h.p., the instance hypergraph generated has \( o(n) \) cycles of length at most \( \eta \log n \) for \( \eta > 0 \).

Remove one constraint from every small cycle and get an instance of girth \( \eta \log n \).
#variables = n and #constraints = m = \Delta \cdot n.

- For each variable in \( \Phi_0 \), create bucket with large number of variables.
- Independently, sample each constraint as:
  - Sample constraint \( C \) from \( \Phi_0 \).
Hard Instance

\#variables = n \quad \text{and} \quad \#constraints = m = \Delta \cdot n.

- For each variable in $\Phi_0$, create bucket with large number of variables.

- Independently, sample each constraint as:
  - Sample constraint $C$ from $\Phi_0$.
  - For each variable $x$ in $S_C$, choose $y_x \in B_x$, u.a.r.
Hard Instance

#variables = n  and  #constraints = m = \Delta \cdot n.

- For each variable in \( \Phi_0 \), create bucket with large number of variables.
- Independently, sample each constraint as:
  - Sample constraint \( C \) from \( \Phi_0 \).
  - For each variable \( x \) in \( S_C \), choose \( y_x \in B_x \), u.a.r.
  - Put the constraint \( C \) on the variables \( \{y_x\}_{x \in S_C} \).
Hard Instance

#variables = n and #constraints = m = Δ \cdot n.

- For each variable in $\Phi_0$, create bucket with large number of variables.
- Independently, sample each constraint as:
  - Sample constraint $C$ from $\Phi_0$.
  - For each variable $x$ in $S_C$, choose $y_x \in B_x$, u.a.r.
  - Put the constraint $C$ on the variables $\{y_x\}_{x \in S_C}$.

W.h.p., the instance hypergraph generated has $o(n)$ cycles of length at most $\eta \log n$ for $\eta > 0$. 
Hard Instance

#variables = n and #constraints = m = \Delta \cdot n.

- For each variable in \( \Phi_0 \), create bucket with large number of variables.
- Independently, sample each constraint as:
  1. Sample constraint \( C \) from \( \Phi_0 \).
  2. For each variable \( x \) in \( S_C \), choose \( y_x \in B_x \), u.a.r.
  3. Put the constraint \( C \) on the variables \( \{y_x\}_{x \in S_C} \).

W.h.p., the instance hypergraph generated has \( o(n) \) cycles of length at most \( \eta \log n \) for \( \eta > 0 \). Remove one constraint from every small cycle and get an instance of girth \( \eta \log n \).
Overview - Completeness

Instance:

Consistent Distributions:

\[ S \cap T \cap D_S \cap D_T \cap D_{S \cap T} \cap T \]
Overview - Completeness

Instance:

Consistent Distributions:

\( S \cap T \)

\( \mathcal{D}_S \)

\( \mathcal{D}_{S \cap T} \)

\( \mathcal{D}_T \)

Step 2: Construction of consistent distribution – Conditioning and propagating.
Overview - Completeness

Instance:

Consistent Distributions:

\[
S \cap T
\]

Step 1: Consistent Low-Diameter Decompositions.

Step 2: Construction of consistent distribution – Conditioning and propagating.
Step 1: Requirements

- A family of distributions, $\{C_S\}_{|S| \leq t}$
Step 1: Requirements

- A family of distributions, \( \{C_S\}_{|S| \leq t} \)
- \( C_S \): a distribution supported on partitions of \( S \) into low-diameter (not necessarily connected) components in the hypergraph.
Step 1: Requirements

- A family of distributions, \( \{C_S\}_{|S| \leq t} \)
- \( C_S \): a distribution supported on partitions of \( S \) into low-diameter (not necessarily connected) components in the hypergraph. Target diameter = \( \text{girth}/100 \).
Step 1: Requirements

- A family of distributions, \( \{C_S\}_{|S| \leq t} \)
- \( C_S \): a distribution supported on partitions of \( S \) into low-diameter (not necessarily connected) components in the hypergraph. Target diameter = girth/100.

- Minimize the quantity: the probability of a hyperedge being cut. Target = \( \varepsilon \).
Step 1: Requirements

- A family of distributions, $\{C_S\}_{|S| \leq t}$
- $C_S$: a distribution supported on partitions of $S$ into low-diameter (not necessarily connected) components in the hypergraph. Target diameter $= \text{girth}/100$.
- Consistency:

Figure: $S \subset T$

- Minimize the quantity: the probability of a hyperedge being cut. Target $= \varepsilon$. 
Step 1: Requirements

- A family of distributions, $\{C_S\}_{|S| \leq t}$
- $C_S$: a distribution supported on partitions of $S$ into low-diameter (not necessarily connected) components in the hypergraph. Target diameter = girth/100.
- Consistency:

Figure: $S \subset T$

- Minimize the quantity: the probability of a hyperedge being cut. Target = $\varepsilon$. 


Step 1: Requirements

- A family of distributions, $\{C_S\}_{|S| \leq t}$
- $C_S$: a distribution supported on partitions of $S$ into \textit{low-diameter (not necessarily connected)} components in the hypergraph. Target diameter = $\text{girth}/100$.

- Consistency:

![Diagram showing a subset relation $S \subset T$]

**Figure:** $S \subset T$

- Minimize the quantity: the probability of a hyperedge being \textbf{cut}. Target = $\varepsilon$. 
Step 1: Requirements

- A family of distributions, $\{C_S\}_{|S|\leq t}$
- $C_S$: a distribution supported on partitions of $S$ into low-diameter (not necessarily connected) components in the hypergraph. Target diameter $= \text{girth}/100$.
- Consistency:

![Diagram showing sets $S$, $T$, $D_S$, $D_S \cap T$, and $D_T$.]

Figure: $S \subset T$

- Minimize the quantity: the probability of a hyperedge being cut. Target $= \varepsilon$. 

Step 2: Conditioning and Propagation

Assume: \( c = 1 \)

Construction of \( \mathcal{D}_S \):

- Sample a partition \( \mathcal{P} \) of \( S \) from \( \mathcal{C}_S \).
Step 2: Conditioning and Propagation

Assume: \( c = 1 \)

Construction of \( \mathcal{D}_S \):
- Sample a partition \( \mathcal{P} \) of \( S \) from \( \mathcal{C}_S \).
- For each cell \( T \) of \( \mathcal{P} \), construct tree \( \mathcal{T}_S \) by connecting all shortest paths. Root the tree arbitrarily.
Step 2: Conditioning and Propagation

Assume: $c = 1$

Construction of $\mathcal{D}_S$:
- Sample a partition $\mathcal{P}$ of $S$ from $\mathcal{C}_S$.
- For each cell $T$ of $\mathcal{P}$, construct tree $\mathcal{T}_S$ by connecting all shortest paths. Root the tree arbitrarily.
- Independently, for each $\mathcal{T}_S$ condition and propagate assignments in $\mathcal{T}_S$ using the local distribution from basic relaxation.

Consistent Distribution.
Step 2: Conditioning and Propagation

Assume: \( c = 1 \)

Construction of \( D_S \):
- Sample a partition \( P \) of \( S \) from \( C_S \).
- For each cell \( T \) of \( P \), construct tree \( T_S \) by connecting all shortest paths. Root the tree arbitrarily.
- Independently, for each \( T_S \) condition and propagate assignments in \( T_S \) using the local distribution from basic relaxation.
Step 2: Conditioning and Propagation

Assume: $c = 1$

Construction of $\mathcal{D}_S$:
- Sample a partition $\mathcal{P}$ of $S$ from $\mathcal{C}_S$.
- For each cell $T$ of $\mathcal{P}$, construct tree $\mathcal{T}_S$ by connecting all shortest paths. Root the tree arbitrarily.
- Independently, for each $\mathcal{T}_S$ condition and propagate assignments in $\mathcal{T}_S$ using the local distribution from basic relaxation.
Step 2: Conditioning and Propagation

Assume: $c = 1$

Construction of $\mathcal{D}_S$:

- Sample a partition $\mathcal{P}$ of $S$ from $\mathcal{C}_S$.
- For each cell $T$ of $\mathcal{P}$, construct tree $\mathcal{T}_S$ by connecting all shortest paths. Root the tree arbitrarily.
- Independently, for each $\mathcal{T}_S$ condition and propagate assignments in $\mathcal{T}_S$ using the local distribution from basic relaxation.
- For cell $T$, retain only the assignments to variables in $T$. 
Step 2: Conditioning and Propagation

Assume: $c = 1$

The cut constraints may not be satisfied.

Construction of $\mathcal{D}_S$:

- Sample a partition $\mathcal{P}$ of $S$ from $\mathcal{C}_S$.
- For each cell $T$ of $\mathcal{P}$, construct tree $\mathcal{T}_S$ by connecting all shortest paths. Root the tree arbitrarily.
- Independently, for each $\mathcal{T}_S$ condition and propagate assignments in $\mathcal{T}_S$ using the local distribution from basic relaxation.
- For cell $T$, retain only the assignments to variables in $T$. 

High girth + consistent low-diameter decomposition $\implies$ Consistent Distribution.
Step 2: Conditioning and Propagation

Assume: $c = 1$

Construction of $\mathcal{D}_S$:

- Sample a partition $\mathcal{P}$ of $S$ from $\mathcal{C}_S$.
- For each cell $T$ of $\mathcal{P}$, construct tree $\mathcal{T}_S$ by connecting all shortest paths. Root the tree arbitrarily.
- Independently, for each $\mathcal{T}_S$ condition and propagate assignments in $\mathcal{T}_S$ using the local distribution from basic relaxation.
- For cell $T$, retain only the assignments to variables in $T$.

The cut constraints may not be satisfied. The distribution for any tree is independent of the choice of root.
Step 2: Conditioning and Propagation

Assume: $c = 1$

The cut constraints may not be satisfied. The distribution for any tree is independent of the choice of root.

High girth + consistent low-diameter decomposition $\Rightarrow$ Consistent Distribution.

Construction of $\mathcal{D}_S$:

- Sample a partition $\mathcal{P}$ of $S$ from $\mathcal{C}_S$.
- For each cell $T$ of $\mathcal{P}$, construct tree $\mathcal{T}_S$ by connecting all shortest paths. Root the tree arbitrarily.
- Independently, for each $\mathcal{T}_S$ condition and propagate assignments in $\mathcal{T}_S$ using the local distribution from basic relaxation.
- For cell $T$, retain only the assignments to variables in $T$. 
Construction of Step 1

Charikar-Makarychev-Makarychev-09: Can define a metric on the hypergraph (that grows with hypergraph distance) so that restriction on any small set is isometrically embeddable on sphere.
Construction of Step 1

Charikar-Makarychev-Makarychev-09: Can define a metric on the hypergraph (that grows with hypergraph distance) so that restriction on any small set is isometrically embeddable on sphere.

Charikar et al. 1998: There exists a rotation invariant, oblivious decomposition of sphere into low diameter components.
Construction of Step 1

Charikar-Makarychev-Makarychev-09: Can define a metric on the hypergraph (that grows with hypergraph distance) so that restriction on any small set is isometrically embeddable on sphere.

Charikar et al. 1998: There exists a rotation invariant, oblivious decomposition of sphere into low diameter components.
Charikar-Makarychev-Makarychev-09: Can define a metric on the hypergraph (that grows with hypergraph distance) so that restriction on any *small* set is *isometrically* embeddable on sphere.

Charikar et al. 1998: There exists a *rotation invariant, oblivious* decomposition of sphere into low diameter components.
Construction of Step 1

Charikar-Makarychev-Makarychev-09: Can define a metric on the hypergraph (that grows with hypergraph distance) so that restriction on any small set is isometrically embeddable on sphere.

Charikar et al. 1998: There exists a rotation invariant, oblivious decomposition of sphere into low diameter components.
Construction of Step 1

Charikar-Makarychev-Makarychev-09: Can define a metric on the hypergraph (that grows with hypergraph distance) so that restriction on any small set is isometrically embeddable on sphere.

Charikar et al. 1998: There exists a rotation invariant, oblivious decomposition of sphere into low diameter components.
Charikar-Makarychev-Makarychev-09: Can define a metric on the hypergraph (that grows with hypergraph distance) so that restriction on any small set is isometrically embeddable on sphere.

Charikar et al. 1998: There exists a rotation invariant, oblivious decomposition of sphere into low diameter components.

The probability of cutting a hyperedge dictates the size of the sets we can handle.
Conclusion

- We prove a dichotomy result for all CSPs for linear programming relaxations.
- We prove a dichotomy result for all CSPs for linear programming relaxations.

- The result can also be interpreted as reducing the problem of showing hardness to a possibly easier task.
Conclusion

- We prove a dichotomy result for all CSPs for linear programming relaxations.

- The result can also be interpreted as reducing the problem of showing hardness to a possibly easier task.

Q: Can the number of levels of $SA$ be improved?
Conclusion

- We prove a dichotomy result for all CSPs for linear programming relaxations.

- The result can also be interpreted as reducing the problem of showing hardness to a possibly easier task.

Q: Can the number of levels of SA be improved?

Q: What can be said for the case of SDP hierarchies?
Conclusion

- We prove a dichotomy result for all CSPs for linear programming relaxations.

- The result can also be interpreted as reducing the problem of showing hardness to a possibly easier task.

Q: Can the number of levels of SA be improved?
Q: What can be said for the case of SDP hierarchies?

Questions?
Other Dichotomy Results

- [Raghavendra-08]: Assuming Unique Games Conjecture, either a basic SDP achieves a \((c, s)\)-approximation for a CSP or it is NP-hard to do so.
Other Dichotomy Results

- **[Raghavendra-08]:** Assuming Unique Games Conjecture, either a basic SDP achieves a \((c, s)\)-approximation for a CSP or it is NP-hard to do so.

- **[Raghavendra-Steurer-09]:** (For Unique Games) If a basic SDP has gap of \((c, s)\) then so does \(\tilde{\log \log n}^{\frac{1}{4}}\)-levels of mixed relaxation.
Other Dichotomy Results

- **[Raghavendra-08]**: Assuming Unique Games Conjecture, either a basic SDP achieves a \((c, s)\)-approximation for a CSP or it is NP-hard to do so (for th

- **[Raghavendra-Steurer-09]**: (For Unique Games) If a basic SDP has gap of \((c, s)\) then so does \((\log \log n)^{\frac{1}{4}}\)-levels of mixed relaxation.

- **This result** If basic LP relaxation has a gap of \((c, s)\), then so does \(\tilde{O}(\log n)\)-level SA.