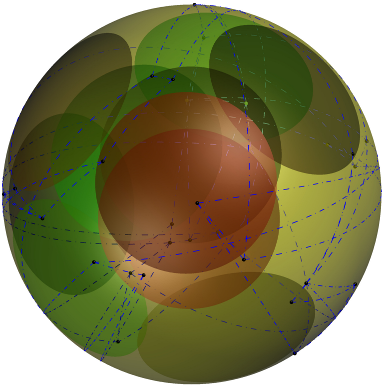
From Weak to Strong LP Gaps for all CSPs



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

$$x_1 \vee x_{22} \vee \overline{x}_{19}$$
$$x_3 \vee \overline{x}_9 \vee x_{23}$$
$$x_5 \vee \overline{x}_7 \vee \overline{x}_9$$
$$\vdots$$

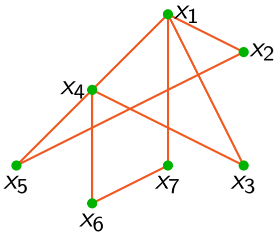
MAX k-CSP

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

Max-3-SAT

$$x_1 \vee x_{22} \vee \overline{x}_{19}$$
$$x_3 \vee \overline{x}_9 \vee x_{23}$$
$$x_5 \vee \overline{x}_7 \vee \overline{x}_9$$
$$\vdots$$

Max-Cut



$$x_1 \neq x_2$$
$$x_2 \neq x_5$$
$$x_3 \neq x_4$$
$$\vdots$$

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

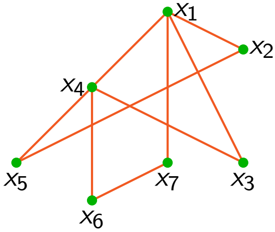
$x_1 \vee x_{22} \vee \overline{x}_{19}$

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

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

$\vdots$

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.

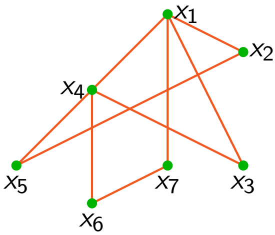
MAX k-CSP

- 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

$$x_1 \vee x_{22} \vee \overline{x}_{19}$$
$$x_3 \vee \overline{x}_9 \vee x_{23}$$
$$x_5 \vee \overline{x}_7 \vee \overline{x}_9$$
$$\vdots$$

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.

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 α to S_{C_i}.

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 α to S_{C_i}.

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

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

$$x_{(S,\alpha)} \geq 0$$

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 α to S_{C_i}.

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

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

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

$$x_{(S,\alpha)} \geq 0$$

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 α to S_{C_i}.

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

$$\sum_{\substack{\alpha \in \{0,1\}^{S_C} \\ \alpha(i) = b}} x_{(S_C, \alpha)} = x_{(i,b)} \qquad \forall C \in \Phi, i \in S_C, \; b \in \{0,1\}$$

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

$$x_{(S,\alpha)} \geq 0$$

#constraints = $\Theta\left(m \cdot 2^k\right)$

Extended Formulation and Sherali-Adams Relaxation

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

Extended Formulation and Sherali-Adams Relaxation

- **Extended Formulation:** Defined by a feasible polytope P, and a way of encoding instances Φ as a (linear) objective function w_Φ.
- Optimize objective $\langle w_\Phi, x \rangle$ (depending on Φ) over P.

Extended Formulation and Sherali-Adams Relaxation

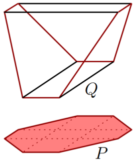


Image from
[Fiorini-Rothvoss-Tiwari-11]

- **Extended Formulation:** Defined by a feasible polytope P, and a way of encoding instances Φ as a (linear) objective function w_Φ.
- Optimize objective $\langle w_\Phi, x \rangle$ (depending on Φ) over P.
- Introduce additional variables y. Optimize over polytope $P = \{x \mid \exists y \;\; Ex + Fy = g, y \geq 0\}$. Size equals #variables + #constraints.

Extended Formulation and Sherali-Adams Relaxation

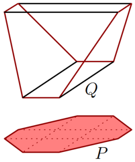


Image from [Fiorini-Rothvoss-Tiwari-11]

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

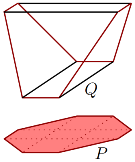
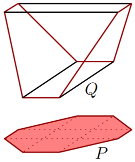


Image from [Fiorini-Rothvoss-Tiwari-11]

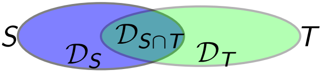
- **Extended Formulation:** Defined by a feasible polytope P, and a way of encoding instances Φ as a (linear) objective function w_Φ.
- **Sherali-Adams:** A Sherali-Adams of level t is an Extended Formulation with #variables $= \binom{n}{t} \cdot 2^t$.
- Variables: $x_{(S,\alpha)}$, $|S| \leq t$, $\alpha \in \{0,1\}^S$.

Extended Formulation and Sherali-Adams Relaxation

EF:



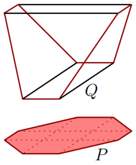
SA:



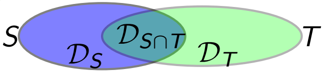
- **Extended Formulation:** Defined by a feasible polytope P, and a way of encoding instances Φ as a (linear) objective function w_Φ.
- **Sherali-Adams:** A Sherali-Adams of level t is an Extended Formulation with #variables $= \binom{n}{t} \cdot 2^t$.
- Feasible point in $SA(t)$: Family $\{\mathcal{D}_S\}_{|S|\leq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

Extended Formulation and Sherali-Adams Relaxation

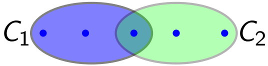
EF:



SA:

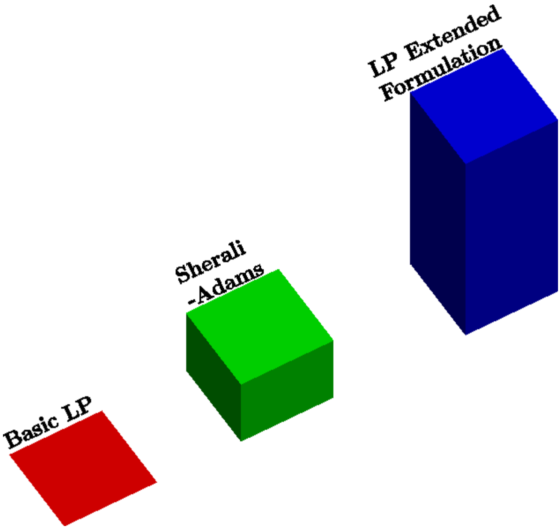


Basic:

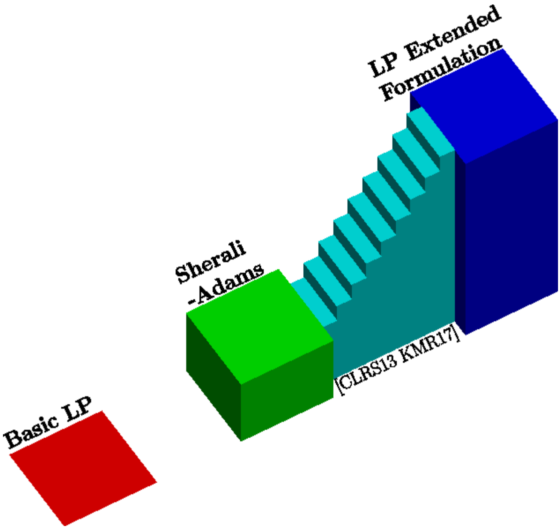


- **Extended Formulation:** Defined by a feasible polytope P, and a way of encoding instances Φ as a (linear) objective function w_Φ.
- **Sherali-Adams:** A Sherali-Adams of level t is an Extended Formulation with $\#\text{variables} = \binom{n}{t} \cdot 2^t$.
- Feasible point in $SA(t)$: Family $\{\mathcal{D}_S\}_{|S|\leq t}$ of consistent distribution with $\mathcal{D}_S$ a distribution on $\{0,1\}^S$.
- Similarly, for Basic LP solution.

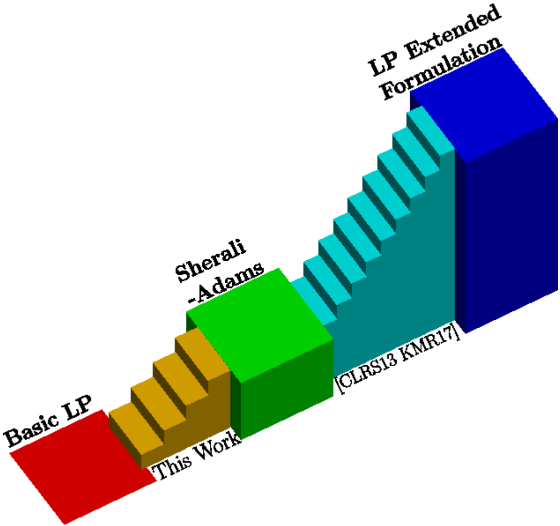
Result



Result

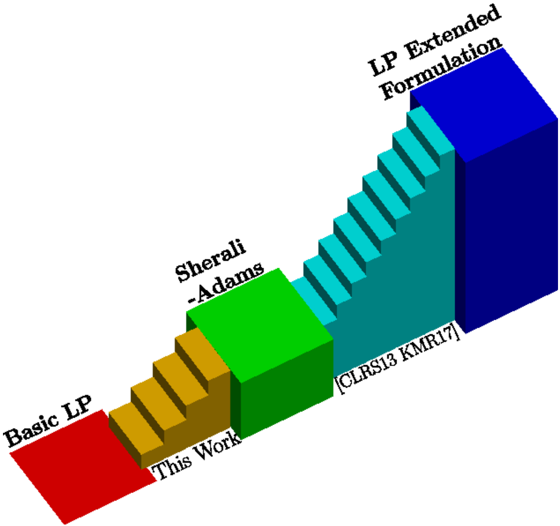

LP Extended
Formulation
Sherali
-Adams
[CLRS13 KMR17]
Basic LP

Result



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 $SA(\widetilde{O}_{\varepsilon}(\log n))$.

Result

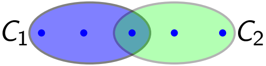


With [Kothari-Meka-Raghavendra-17]: **For *all* CSPs, if Basic LP has (c, s) gap, then so does any LP Extended Formulation of size $n^{\widetilde{O}(\log n)}$.**

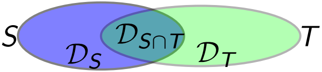
Ignoring ε losses.

Hard Instance

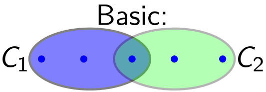
Basic:

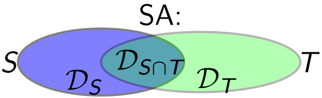


SA:



Hard Instance





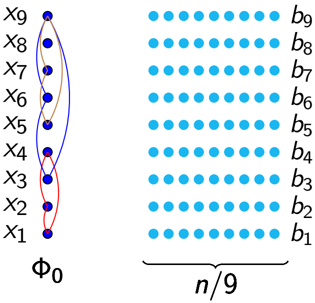
Use the hard instance Φ_0 of the basic relaxation as template to build the new hard instance on n variables and $m = \Delta \cdot n$ constraints.

Hard Instance

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

Hard Instance

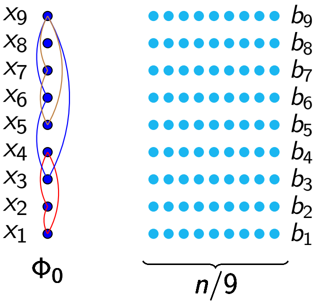
$\#\text{variables} = n$ and $\#\text{constraints} = m = \Delta \cdot n.$



- For each variable in Φ_0, create bucket with large number of variables.

Hard Instance

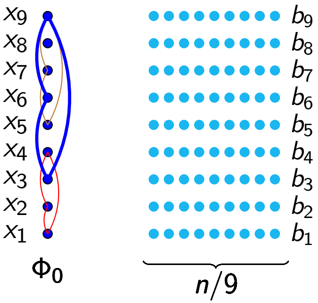
$\#\text{variables} = n$ and $\#\text{constraints} = m = \Delta \cdot n.$



- For each variable in Φ_0, create bucket with large number of variables.
- Independently, sample each constraint as:

Hard Instance

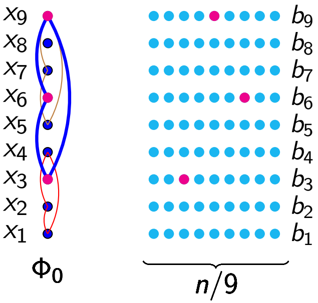
$$\#\text{variables} = n \quad \text{and} \quad \#\text{constraints} = m = \Delta \cdot n.$$



- For each variable in Φ_0, create bucket with large number of variables.
- Independently, sample each constraint as:
  - Sample constraint C from Φ_0.

Hard Instance

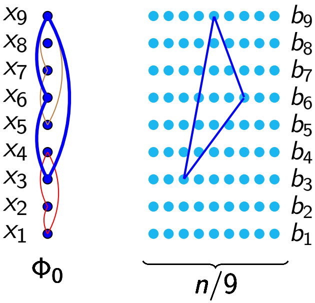
$\#\text{variables} = n$ and $\#\text{constraints} = m = \Delta \cdot n$.



- For each variable in Φ_0, create bucket with large number of variables.
- Independently, sample each constraint as:
  - Sample constraint C from Φ_0.
  - For each variable x in S_C, choose $y_x \in B_x$, u.a.r.

Hard Instance

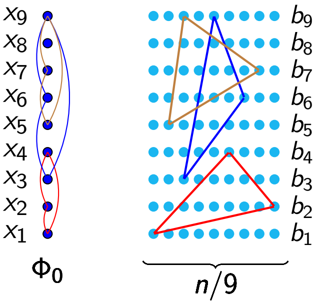
$\#\text{variables} = n$ and $\#\text{constraints} = m = \Delta \cdot n.$



- For each variable in Φ_0, create bucket with large number of variables.
- Independently, sample each constraint as:
  - Sample constraint C from Φ_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

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

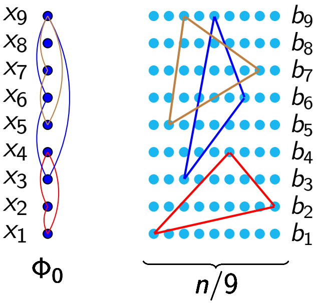


- For each variable in Φ_0, create bucket with large number of variables.
- Independently, sample each constraint as:
  - Sample constraint C from Φ_0.
  - For each variable x in S_C, choose $y_x \in B_x$, u.a.r.
  - Put the constraint C on the variables $\{y_x\}_{x \in S_C}$.

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

Hard Instance

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

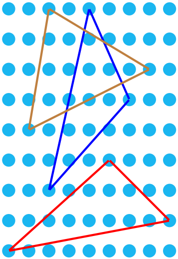


- For each variable in Φ_0, create bucket with large number of variables.
- Independently, sample each constraint as:
  - Sample constraint C from Φ_0.
  - For each variable x in S_C, choose $y_x \in B_x$, u.a.r.
  - Put the constraint C on the variables $\{y_x\}_{x \in S_C}$.

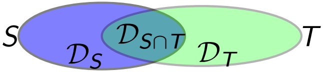
W.h.p., the instance hypergraph generated has $o(n)$ cycles of length at most $\eta \log n$ for $\eta > 0$. Remove one constraint from every small cycle and get an instance of girth $\eta \log n$.

Overview - Completeness

Instance:

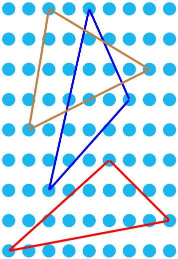


Consistent Distributions:

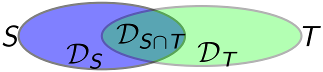


Overview - Completeness

Instance:



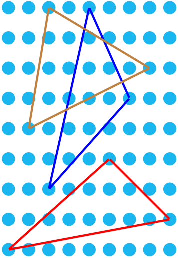
Consistent Distributions:



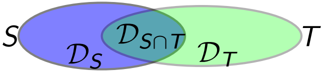
Step 2: Construction of consistent distribution – Conditioning and propagating.

Overview - Completeness

Instance:



Consistent Distributions:



Step 1: Consistent Low-Diameter Decompositions.

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

Step 1: Requirements

- A family of distributions, $\{\mathcal{C}_S\}_{|S| \leq t}$

Step 1: Requirements

- A family of distributions, $\{\mathcal{C}_S\}_{|S| \leq t}$
- $\mathcal{C}_S$: a distribution supported on partitions of S into *low-diameter* (not necessarily connected) components in the hypergraph.

Step 1: Requirements

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

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

Step 1: Requirements

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

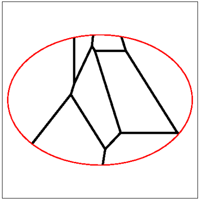


Figure: $S \subset T$

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

Step 1: Requirements

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

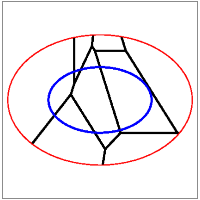


Figure: $S \subset T$

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

Step 1: Requirements

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

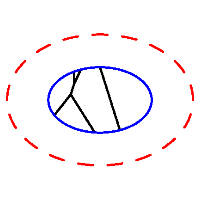


Figure: $S \subset T$

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

Step 1: Requirements

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

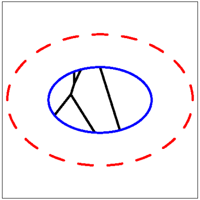
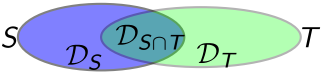


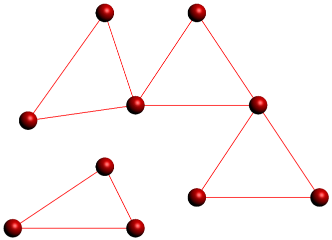
Figure: $S \subset T$



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

Step 2: Conditioning and Propagation

Assume: $c = 1$

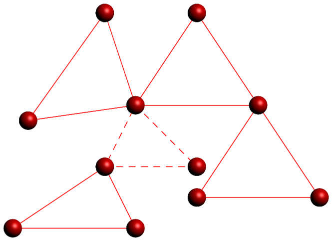


Construction of $\mathcal{D}_S$:

- Sample a partition $\mathcal{P}$ of S from $\mathcal{C}_S$.

Step 2: Conditioning and Propagation

Assume: $c = 1$

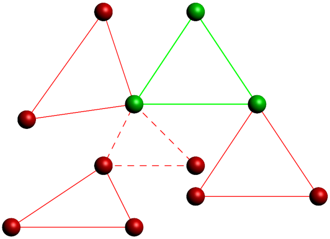


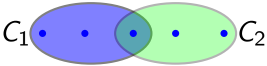
Construction of $\mathcal{D}_S$:

- 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: $c = 1$



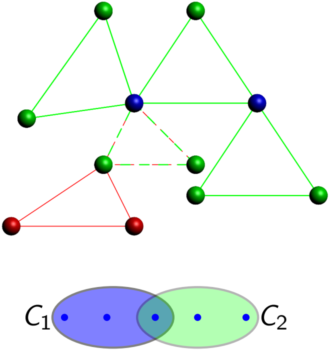


Construction of $\mathcal{D}_S$:

- 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.

Step 2: Conditioning and Propagation

Assume: $c = 1$

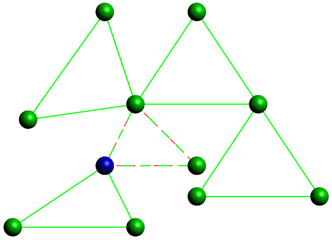


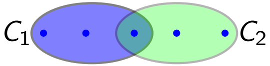
Construction of $\mathcal{D}_S$:

- 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: $c = 1$



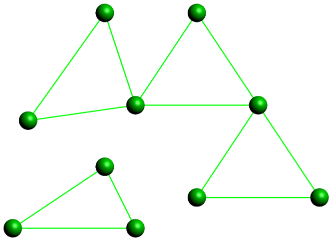


Construction of $\mathcal{D}_S$:

- 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.

Step 2: Conditioning and Propagation

Assume: $c = 1$

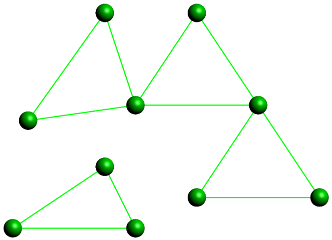


Construction of $\mathcal{D}_S$:

- 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: $c = 1$



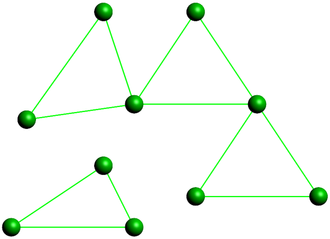
The cut constraints may not be satisfied.

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.

Step 2: Conditioning and Propagation

Assume: $c = 1$



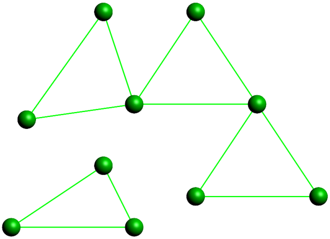
The cut constraints may not be satisfied. The distribution for any tree is independent of the choice of root.

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: $c = 1$



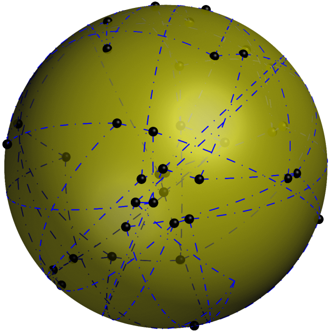
The cut constraints may not be satisfied. The distribution for any tree is independent of the choice of root.

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.

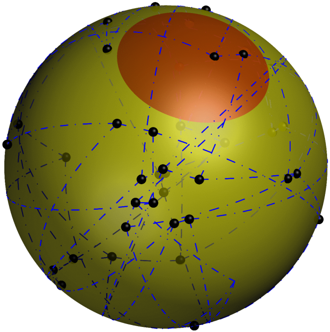
High girth + consistent low-diameter decomposition $\Rightarrow$ Consistent Distribution.

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.

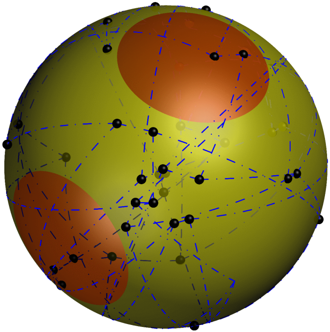
Construction of Step 1



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.

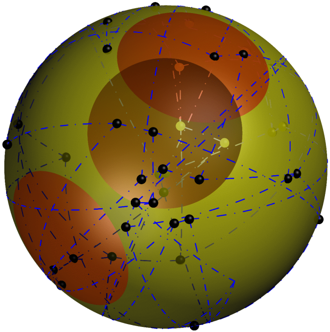
Construction of Step 1



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.

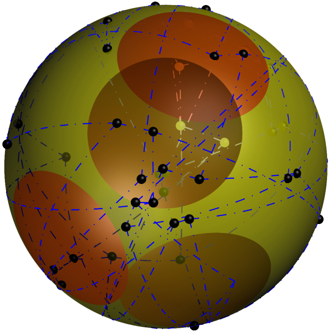
Construction of Step 1



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.

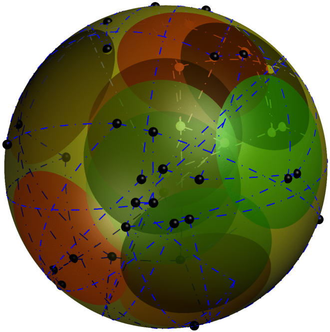
Construction of Step 1



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.

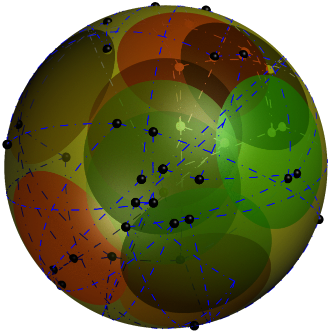
Construction of Step 1



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.

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.

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.

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 (for th

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.

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 $\widetilde{O}(\log n)$-level SA.