Robust satisfiability of CSPs

Libor Barto (Charles University in Prague)

joint work with Marcin Kozik (Jagiellonian University)

Dagstuhl, November 4, 2012

Constraint Satisfaction Problem (CSP)

Definition (Instance of the CSP)

Instance of the CSP consists of:

- V ... a set of **variables**
- A ... a **domain**
- list of **constraints** of the form $R(x_1, \dots, x_k)$, where
  - $x_1, \dots, x_k \in V$
  - R is a k-ary relation on A (i.e. $R \subseteq A^k$) **constraint relation**

Constraint Satisfaction Problem (CSP)

Definition (Instance of the CSP)

Instance of the CSP consists of:

- V ... a set of **variables**
- A ... a **domain**
- list of **constraints** of the form $R(x_1, \dots, x_k)$, where
  - $x_1, \dots, x_k \in V$
  - R is a k-ary relation on A (i.e. $R \subseteq A^k$) **constraint relation**

An assignment $f : V \to A$ satisfies $R(x_1, \dots, x_k)$, if $(f(x_1), \dots, f(x_k)) \in R$

$f : V \to A$ is a solution if it satisfies all the constraints

Some questions we can ask

- **Decision CSP:** Does a solution exist?
- **Max-CSP:** Find a map satisfying maximum number of constraints
- **Approx. Max-CSP:** Find a map satisfying at least $0.7 \times \mathit{Optimum}$ constraints

Some questions we can ask

- **Decision CSP:** Does a solution exist?
- **Max-CSP:** Find a map satisfying maximum number of constraints
- **Approx. Max-CSP:** Find a map satisfying at least $0.7 \times$ *Optimum* constraints

Definition

An algorithm (α, β)-approximates CSP $(0 \leq \alpha \leq \beta \leq 1)$ if it returns an assignment satisfying α-fraction of the constraints given a β-satisfiable instance.

Some questions we can ask

- **Decision CSP:** Does a solution exist?
- **Max-CSP:** Find a map satisfying maximum number of constraints
- **Approx. Max-CSP:** Find a map satisfying at least $0.7 \times$ *Optimum* constraints

Definition

An algorithm (α, β)-approximates CSP $(0 \leq \alpha \leq \beta \leq 1)$ if it returns an assignment satisfying α-fraction of the constraints given a β-satisfiable instance.

Example

$(0.7\beta, \beta)$-approximating algorithm returns a map satisfying at least $0.7 \times$ *Optimum* constraints.

Fixed constraint language

Definition

A constraint language Γ is a finite set of relations on a finite set A. An instance of $\mathrm{CSP}(\Gamma)$ is a CSP instance such that every constraint relation is from Γ.

Fixed constraint language

Definition

A constraint language Γ is a finite set of relations on a finite set A. An instance of $\mathrm{CSP}(\Gamma)$ is a CSP instance such that every constraint relation is from Γ.

Given Γ:

- Is decision CSP(Γ) in P? $=$ Is $(1, 1)$-approximation in P?
- For which α, β is (α, β)-approximation of CSP(Γ) in P
- In between: Is robust approximation of CSP(Γ) in P?

Between decision and approximation

Definition (Zwick'98)

$\mathrm{CSP}(\Gamma)$ admits a robust algorithm, if there is a polynomial time algorithm which
$(1-g(\varepsilon), 1-\varepsilon)$-approximates $\mathrm{CSP}(\Gamma)$ (for every ε),
where $g(\varepsilon) \to 0$ when $\varepsilon \to 0$, and $g(0) = 0$.

Between decision and approximation

Definition (Zwick'98)

$\mathrm{CSP}(\Gamma)$ admits a robust algorithm, if there is a polynomial time algorithm which
$(1 - g(\varepsilon), 1 - \varepsilon)$-approximates $\mathrm{CSP}(\Gamma)$ (for every ε),
where $g(\varepsilon) \to 0$ when $\varepsilon \to 0$, and $g(0) = 0$.

Motivation: Instances close to satisfiable (e.g. corrupted by noise), we want to find an "almost solution".

Between decision and approximation

Definition (Zwick'98)

$\mathrm{CSP}(\Gamma)$ admits a robust algorithm, if there is a polynomial time algorithm which
$(1 - g(\varepsilon), 1 - \varepsilon)$-approximates $\mathrm{CSP}(\Gamma)$ (for every ε),
where $g(\varepsilon) \to 0$ when $\varepsilon \to 0$, and $g(0) = 0$.

Motivation: Instances close to satisfiable (e.g. corrupted by noise), we want to find an "almost solution".

Techniques: Linear programming (LP), Semidefinite programming (SDP)

Between decision and approximation

Definition (Zwick'98)

$\mathrm{CSP}(\Gamma)$ admits a robust algorithm, if there is a polynomial time algorithm which
$(1-g(\varepsilon), 1-\varepsilon)$-approximates $\mathrm{CSP}(\Gamma)$ (for every ε),
where $g(\varepsilon) \to 0$ when $\varepsilon \to 0$, and $g(0) = 0$.

Motivation: Instances close to satisfiable (e.g. corrupted by noise), we want to find an "almost solution".

Techniques: Linear programming (LP), Semidefinite programming (SDP)

Questions:

- For which Γ does $\mathrm{CSP}(\Gamma)$ admit a robust algorithm?
- What is (asymptotically) the best dependence of g on ε?

Positive results

- HORN-k-SAT
  - $(1 - O(1/(\log(1/\varepsilon))), 1 - \varepsilon)$ **LP** Zwick'98
- HORN-2-SAT
  - $(1 - 3\varepsilon, 1 - \varepsilon)$ Khanna, Sudan, Trevisan, Williamson'00
  - $(1 - 2\varepsilon, 1 - \varepsilon)$ Guruswami, Zhou'11
- 2-SAT
  - $(1 - O(\varepsilon^{1/3}), 1 - \varepsilon)$ **SDP** Zwick'98
  - $(1 - O(\varepsilon^{1/2}), 1 - \varepsilon)$ Charikar, 2 $\times$ Makarychev'09
  - the same bound for CUT Goemans, Williamson'95
- Unique-Games(q) - generalization of CUT
  - $(1 - O(\varepsilon^{1/5} \log^{1/2}(1/\varepsilon)), 1 - \varepsilon)$ Khot'02
  - $(1 - O(\varepsilon^{1/2}), 1 - \varepsilon)$ Charikar, 2 $\times$ Makarychev'06

Positive results

- HORN-k-SAT
  - $(1 - O(1/(\log(1/\varepsilon))), 1 - \varepsilon)$ **LP** Zwick'98
- HORN-2-SAT
  - $(1 - 3\varepsilon, 1 - \varepsilon)$ Khanna, Sudan, Trevisan, Williamson'00
  - $(1 - 2\varepsilon, 1 - \varepsilon)$ Guruswami, Zhou'11
- 2-SAT
  - $(1 - O(\varepsilon^{1/3}), 1 - \varepsilon)$ **SDP** Zwick'98
  - $(1 - O(\varepsilon^{1/2}), 1 - \varepsilon)$ Charikar, 2 $\times$ Makarychev'09
  - the same bound for CUT Goemans, Williamson'95
- Unique-Games(q) - generalization of CUT
  - $(1 - O(\varepsilon^{1/5} \log^{1/2}(1/\varepsilon)), 1 - \varepsilon)$ Khot'02
  - $(1 - O(\varepsilon^{1/2}), 1 - \varepsilon)$ Charikar, 2 $\times$ Makarychev'06

Essentially optimal assuming UGC Khot'02, Khot, Kindler, Mossel, O'Donnell'07, Guruswami, Zhou'11

Negative results

- If the decision $\mathrm{CSP}(\Gamma)$ is NP-complete, then $\mathrm{CSP}(\Gamma)$ has no robust algorithm
  - PCP theorem for $|A| = 2$ Khanna,Sudan,Trevisan, Williamson'00
  - for larger A Jonsson, Krokhin, Kuivinen'09

Negative results

- If the decision $\mathrm{CSP}(\Gamma)$ is NP-complete, then $\mathrm{CSP}(\Gamma)$ has no robust algorithm
  - PCP theorem for $|A| = 2$ Khanna,Sudan,Trevisan, Williamson'00
  - for larger A Jonsson, Krokhin, Kuivinen'09
- LIN-p has no robust algorithm Hastad'01

Negative results

- If the decision $\mathrm{CSP}(\Gamma)$ is NP-complete, then $\mathrm{CSP}(\Gamma)$ has no robust algorithm
  - PCP theorem for $|A| = 2$ Khanna,Sudan,Trevisan, Williamson'00
  - for larger A Jonsson, Krokhin, Kuivinen'09
- LIN-p has no robust algorithm Hastad'01

What distinguishes between
LIN-p, 3-SAT and 2-SAT, HORN-SAT?

Decision CSPs and bounded width

- $\operatorname{Pol}\Gamma =$ clone of polymorphisms (operations compatible with all relations in Γ)
- Complexity of the decision problem for $\mathrm{CSP}(\Gamma)$ controlled by $\mathsf{HSP}(\operatorname{Pol}\Gamma)$ Bulatov, Jeavons, Krokhin 00

Decision CSPs and bounded width

- $\mathsf{Pol}\,\Gamma =$ clone of polymorphisms (operations compatible with all relations in Γ)
- Complexity of the decision problem for $\mathrm{CSP}(\Gamma)$ controlled by $\mathsf{HSP}(\mathsf{Pol}\,\Gamma)$ Bulatov, Jeavons, Krokhin 00
- $\mathrm{CSP}(\Gamma)$ has bounded width iff it can be solved by local consistency checking

Decision CSPs and bounded width

- $\operatorname{Pol}\Gamma =$ clone of polymorphisms (operations compatible with all relations in Γ)
- Complexity of the decision problem for $\mathrm{CSP}(\Gamma)$ controlled by $\mathsf{HSP}(\operatorname{Pol}\Gamma)$ Bulatov, Jeavons, Krokhin 00
- $\mathrm{CSP}(\Gamma)$ has bounded width iff it can be solved by local consistency checking
- $\mathrm{CSP}(\Gamma)$ has bounded width iff Γ "cannot encode linear equations", equivalently, $\mathsf{HSP}(\operatorname{Pol}\Gamma)$ does not contain a reduct of a module (for core Γ) Barto, Kozik'09 Bulatov'09

Decision CSPs and bounded width

- Pol Γ = clone of polymorphisms (operations compatible with all relations in Γ)
- Complexity of the decision problem for $\mathrm{CSP}(\Gamma)$ controlled by HSP(Pol Γ) Bulatov, Jeavons, Krokhin 00
- $\mathrm{CSP}(\Gamma)$ has bounded width iff it can be solved by local consistency checking
- $\mathrm{CSP}(\Gamma)$ has bounded width iff Γ "cannot encode linear equations", equivalently, HSP(Pol Γ) does not contain a reduct of a module (for core Γ) Barto, Kozik'09 Bulatov'09
- Lin-p, 3-SAT do not have bounded width,
  2-SAT, HORN-SAT have bounded width

Decision CSPs and bounded width

- $\operatorname{Pol}\Gamma$ = clone of polymorphisms (operations compatible with all relations in Γ)
- Complexity of the decision problem for $\mathrm{CSP}(\Gamma)$ controlled by $\mathsf{HSP}(\operatorname{Pol}\Gamma)$ Bulatov, Jeavons, Krokhin 00
- $\mathrm{CSP}(\Gamma)$ has bounded width iff it can be solved by local consistency checking
- $\mathrm{CSP}(\Gamma)$ has bounded width iff Γ "cannot encode linear equations", equivalently, $\mathsf{HSP}(\operatorname{Pol}\Gamma)$ does not contain a reduct of a module (for core Γ) Barto, Kozik'09 Bulatov'09
- **Lin-p, 3-SAT do not have bounded width, 2-SAT, HORN-SAT have bounded width !!!**

Decision CSPs and bounded width

- $\mathsf{Pol}\,\Gamma$ = clone of polymorphisms (operations compatible with all relations in Γ)
- Complexity of the decision problem for $\mathrm{CSP}(\Gamma)$ controlled by $\mathsf{HSP}(\mathsf{Pol}\,\Gamma)$ Bulatov, Jeavons, Krokhin 00
- $\mathrm{CSP}(\Gamma)$ has bounded width iff it can be solved by local consistency checking
- $\mathrm{CSP}(\Gamma)$ has bounded width iff Γ "cannot encode linear equations", equivalently, $\mathsf{HSP}(\mathsf{Pol}\,\Gamma)$ does not contain a reduct of a module (for core Γ) Barto, Kozik'09 Bulatov'09
- **Lin-p, 3-SAT do not have bounded width, 2-SAT, HORN-SAT have bounded width**

Conjecture (Guruswami-Zhou 11)

$\mathrm{CSP}(\Gamma)$ admits a robust algorithm iff $\mathrm{CSP}(\Gamma)$ has bounded width.

Universal algebra attacks robust approximation

- robust approximation also (+-) controlled by polymorphisms Dalmau, Krokhin'11
- $\Rightarrow$ one direction of the Guruswami-Zhou conjecture is true

Universal algebra attacks robust approximation

- robust approximation also (+-) controlled by polymorphisms Dalmau, Krokhin'11
- $\Rightarrow$ one direction of the Guruswami-Zhou conjecture is true
- Conjecture confirmed for width 1 CSPs
  Kun, O'Donell, Tamaki, Yoshida, Zhou'11,
  Dalmau, Krokhin'11.
  width 1 iff linear programming relaxation can be used.

Universal algebra attacks robust approximation

- robust approximation also (+-) controlled by polymorphisms Dalmau, Krokhin'11
- $\Rightarrow$ one direction of the Guruswami-Zhou conjecture is true
- Conjecture confirmed for width 1 CSPs Kun, O'Donell, Tamaki, Yoshida, Zhou'11, Dalmau, Krokhin'11.
  width 1 iff linear programming relaxation can be used.
- **Conjecture confirmed Barto, Kozik'11.** Using a semidefinite programming relaxation and Prague strategies.
  - Randomized $(1 - O(\log\log(1/\varepsilon)/\log(1/\varepsilon)), 1-\varepsilon)$-approx algorithm
  - Deterministic $(1 - O(\log\log(1/\varepsilon)/\sqrt{\log(1/\varepsilon)}), 1-\varepsilon)$-approx algorithm

Universal algebra attacks robust approximation

- robust approximation also (+-) controlled by polymorphisms Dalmau, Krokhin'11
- $\Rightarrow$ one direction of the Guruswami-Zhou conjecture is true
- Conjecture confirmed for width 1 CSPs Kun, O'Donell, Tamaki, Yoshida, Zhou'11, Dalmau, Krokhin'11.
  width 1 iff linear programming relaxation can be used.
- **Conjecture confirmed Barto, Kozik'11.** Using a semidefinite programming relaxation and Prague strategies.
  - Randomized $(1 - O(\log\log(1/\varepsilon)/\log(1/\varepsilon)), 1-\varepsilon)$-approx algorithm
  - Deterministic $(1 - O(\log\log(1/\varepsilon)/\sqrt{\log(1/\varepsilon)}), 1-\varepsilon)$-approx algorithm
- Krokhin'11: even the quantitative dependence on ε is +-controlled by polymorphisms.

SDP relaxation for general CSP

Notation and simplifying assumptions:

- A: domain
- Γ contains only binary relations, $\mathrm{CSP}(\Gamma)$ has bounded width
- V: variables, $\mathcal{I}$: instance, $\mathcal{C}$: constraints

SDP relaxation for general CSP

Notation and simplifying assumptions:

- A: domain
- Γ contains only binary relations, $\mathrm{CSP}(\Gamma)$ has bounded width
- V: variables, $\mathcal{I}$: instance, $\mathcal{C}$: constraints
- $\forall \{x, y\} \subseteq V$, $x \neq y$ there is at most one constraint $R_{xy}(x, y) \in \mathcal{C}$

SDP relaxation for general CSP

Notation and simplifying assumptions:

- A: domain
- Γ contains only binary relations, $\mathrm{CSP}(\Gamma)$ has bounded width
- V: variables, $\mathcal{I}$: instance, $\mathcal{C}$: constraints
- $\forall \{x, y\} \subseteq V$, $x \neq y$ there is at most one constraint $R_{xy}(x, y) \in \mathcal{C}$
- $\mathrm{Opt}(\mathcal{I}) = 1 - \varepsilon$: optimal fraction of satisfied constraints
- We want to find an assignment satisfying almost all constraints)

SDP relaxation for general CSP

Notation and simplifying assumptions:

- A: domain
- Γ contains only binary relations, $\mathrm{CSP}(\Gamma)$ has bounded width
- V: variables, $\mathcal{I}$: instance, $\mathcal{C}$: constraints
- $\forall \{x, y\} \subseteq V$, $x \neq y$ there is at most one constraint $R_{xy}(x, y) \in \mathcal{C}$
- $\mathrm{Opt}(\mathcal{I}) = 1 - \varepsilon$: optimal fraction of satisfied constraints
- We want to find an assignment satisfying almost all constraints)

Canonical SDP relaxation is strong enough to get optimal approximation constants (assuming UGC) Raghavendra'08

Canonical SDP relaxation

Find vectors $g(x, a) =: \mathbf{x}_a, x \in V, a \in A$ (notation: $\mathbf{x}_B = \sum_{a \in B} \mathbf{x}_a$)

Canonical SDP relaxation

Find vectors $g(x, a) =: \mathbf{x}_a, x \in V, a \in A$ (notation: $\mathbf{x}_B = \sum_{a \in B} \mathbf{x}_a$)
such that for all $x, y \in V$, $a, b \in A$

- (SDP1) $\mathbf{x}_a \mathbf{y}_b \geq 0$
- (SDP2) $\mathbf{x}_a \mathbf{x}_b = 0$ if $a \neq b$
- (SDP3) $\mathbf{x}_A = \mathbf{y}_A$, $\|\mathbf{x}_A\|^2 = 1$

Canonical SDP relaxation

Find vectors $g(x, a) =: \mathbf{x}_a, x \in V, a \in A$ (notation: $\mathbf{x}_B = \sum_{a \in B} \mathbf{x}_a$)
such that for all $x, y \in V$, $a, b \in A$

- (SDP1) $\mathbf{x}_a \mathbf{y}_b \geq 0$
- (SDP2) $\mathbf{x}_a \mathbf{x}_b = 0$ if $a \neq b$
- (SDP3) $\mathbf{x}_A = \mathbf{y}_A$, $\|\mathbf{x}_A\|^2 = 1$

maximizing

$$\mathrm{SDPOpt}(\mathcal{I}) = \frac{1}{|\mathcal{C}|} \sum_{R_{xy}(x,y) \in \mathcal{C}} \sum_{(a,b) \in R_{xy}} \mathbf{x}_a \mathbf{y}_b.$$

Canonical SDP relaxation

Find vectors $g(x, a) =: \mathbf{x}_a, x \in V, a \in A$ (notation: $\mathbf{x}_B = \sum_{a \in B} \mathbf{x}_a$)
such that for all $x, y \in V$, $a, b \in A$

- (SDP1) $\mathbf{x}_a \mathbf{y}_b \geq 0$
- (SDP2) $\mathbf{x}_a \mathbf{x}_b = 0$ if $a \neq b$
- (SDP3) $\mathbf{x}_A = \mathbf{y}_A$, $\|\mathbf{x}_A\|^2 = 1$

maximizing

$$\mathrm{SDPOpt}(\mathcal{I}) = \frac{1}{|\mathcal{C}|} \sum_{R_{xy}(x,y) \in \mathcal{C}} \sum_{(a,b) \in R_{xy}} \mathbf{x}_a \mathbf{y}_b.$$

Intuition:

- $\mathbf{x}_a \mathbf{y}_b$ is the weight (nonnegative) of the pair (a, b) between variables x, y

Canonical SDP relaxation

Find vectors $g(x, a) =: \mathbf{x}_a, x \in V, a \in A$ (notation: $\mathbf{x}_B = \sum_{a \in B} \mathbf{x}_a$)
such that for all $x, y \in V$, $a, b \in A$

- (SDP1) $\mathbf{x}_a \mathbf{y}_b \geq 0$
- (SDP2) $\mathbf{x}_a \mathbf{x}_b = 0$ if $a \neq b$
- (SDP3) $\mathbf{x}_A = \mathbf{y}_A$, $\|\mathbf{x}_A\|^2 = 1$

maximizing

$$\mathrm{SDPOpt}(\mathcal{I}) = \frac{1}{|\mathcal{C}|} \sum_{R_{xy}(x,y) \in \mathcal{C}} \sum_{(a,b) \in R_{xy}} \mathbf{x}_a \mathbf{y}_b.$$

Intuition:

- $\mathbf{x}_a \mathbf{y}_b$ is the weight (nonnegative) of the pair (a, b) between variables x, y
- Sum of all weights (between x, y) is 1 from (SDP3)

Canonical SDP relaxation

Find vectors $g(x,a) =: \mathbf{x}_a, x \in V, a \in A$ (notation: $\mathbf{x}_B = \sum_{a \in B} \mathbf{x}_a$)
such that for all $x, y \in V$, $a, b \in A$

- (SDP1) $\mathbf{x}_a \mathbf{y}_b \geq 0$
- (SDP2) $\mathbf{x}_a \mathbf{x}_b = 0$ if $a \neq b$
- (SDP3) $\mathbf{x}_A = \mathbf{y}_A$, $\|\mathbf{x}_A\|^2 = 1$

maximizing

$$\mathrm{SDPOpt}(\mathcal{I}) = \frac{1}{|\mathcal{C}|} \sum_{R_{xy}(x,y) \in \mathcal{C}} \sum_{(a,b) \in R_{xy}} \mathbf{x}_a \mathbf{y}_b.$$

Intuition:

- $\mathbf{x}_a \mathbf{y}_b$ is the weight (nonnegative) of the pair (a, b) between variables x, y
- Sum of all weights (between x, y) is 1 from (SDP3)
- We are trying to give small weights to pairs outside R_{xy}

Canonical SDP relaxation

Find vectors $g(x,a) =: \mathbf{x}_a, x \in V, a \in A$ (notation: $\mathbf{x}_B = \sum_{a\in B} \mathbf{x}_a$)
such that for all $x, y \in V$, $a, b \in A$

- (SDP1) $\mathbf{x}_a \mathbf{y}_b \geq 0$
- (SDP2) $\mathbf{x}_a \mathbf{x}_b = 0$ if $a \neq b$
- (SDP3) $\mathbf{x}_A = \mathbf{y}_A$, $\|\mathbf{x}_A\|^2 = 1$

maximizing

$$\mathrm{SDPOpt}(\mathcal{I}) = \frac{1}{|\mathcal{C}|} \sum_{R_{xy}(x,y)\in\mathcal{C}} \sum_{(a,b)\in R_{xy}} \mathbf{x}_a \mathbf{y}_b.$$

Always $\mathrm{SDPOpt}(\mathcal{I}) \geq \mathrm{Opt}(\mathcal{I})$

Canonical SDP relaxation

Find vectors $g(x, a) =: \mathbf{x}_a, x \in V, a \in A$ (notation: $\mathbf{x}_B = \sum_{a \in B} \mathbf{x}_a$)
such that for all $x, y \in V$, $a, b \in A$

- (SDP1) $\mathbf{x}_a \mathbf{y}_b \geq 0$
- (SDP2) $\mathbf{x}_a \mathbf{x}_b = 0$ if $a \neq b$
- (SDP3) $\mathbf{x}_A = \mathbf{y}_A$, $\|\mathbf{x}_A\|^2 = 1$

maximizing

$$\mathrm{SDPOpt}(\mathcal{I}) = \frac{1}{|\mathcal{C}|} \sum_{R_{xy}(x,y) \in \mathcal{C}} \sum_{(a,b) \in R_{xy}} \mathbf{x}_a \mathbf{y}_b.$$

Further properties:

- $||\mathbf{x}_a||^2$ is the weight of a

Canonical SDP relaxation

Find vectors $g(x, a) =: \mathbf{x}_a, x \in V, a \in A$ (notation: $\mathbf{x}_B = \sum_{a \in B} \mathbf{x}_a$)
such that for all $x, y \in V$, $a, b \in A$

- (SDP1) $\mathbf{x}_a \mathbf{y}_b \geq 0$
- (SDP2) $\mathbf{x}_a \mathbf{x}_b = 0$ if $a \neq b$
- (SDP3) $\mathbf{x}_A = \mathbf{y}_A$, $\|\mathbf{x}_A\|^2 = 1$

maximizing

$$\mathrm{SDPOpt}(\mathcal{I}) = \frac{1}{|\mathcal{C}|} \sum_{R_{xy}(x,y) \in \mathcal{C}} \sum_{(a,b) \in R_{xy}} \mathbf{x}_a \mathbf{y}_b .$$

Further properties:

- $||\mathbf{x}_a||^2$ is the weight of a
- $||\mathbf{x}_a||^2 =_{(SDP2)} \mathbf{x}_a \mathbf{x}_A =_{(SDP3)} \mathbf{x}_a \mathbf{y}_A$

Canonical SDP relaxation

Find vectors $g(x,a) =: \mathbf{x}_a, x \in V, a \in A$ (notation: $\mathbf{x}_B = \sum_{a \in B} \mathbf{x}_a$)
such that for all $x, y \in V$, $a, b \in A$

- (SDP1) $\mathbf{x}_a \mathbf{y}_b \geq 0$
- (SDP2) $\mathbf{x}_a \mathbf{x}_b = 0$ if $a \neq b$
- (SDP3) $\mathbf{x}_A = \mathbf{y}_A$, $\|\mathbf{x}_A\|^2 = 1$

maximizing

$$\mathrm{SDPOpt}(\mathcal{I}) = \frac{1}{|\mathcal{C}|} \sum_{R_{xy}(x,y) \in \mathcal{C}} \sum_{(a,b) \in R_{xy}} \mathbf{x}_a \mathbf{y}_b.$$

Further properties:

- $||\mathbf{x}_a||^2$ is the weight of a
- $||\mathbf{x}_a||^2 =_{(SDP2)} \mathbf{x}_a \mathbf{x}_A =_{(SDP3)} \mathbf{x}_a \mathbf{y}_A$
- $\Rightarrow$ for every y,
  $||\mathbf{x}_a||^2 =$ sum of weights of edges between x and y via a

Strategy

- We try to produce a good assignment from the SDP output vectors.

Strategy

- We try to produce a good assignment from the SDP output vectors.
- In particular, is it true that
  if $\mathrm{SDPOpt}(\mathcal{I}) = 1$ then $\mathcal{I}$ has a solution?
  This was suggested by Guruswami as the first step to attack the conjecture

Strategy

- We try to produce a good assignment from the SDP output vectors.
- In particular, is it true that
  if $\mathrm{SDPOpt}(\mathcal{I}) = 1$ then $\mathcal{I}$ has a solution?
  This was suggested by Guruswami as the first step to attack the conjecture
- So, assume $\mathrm{SDPOpt}(\mathcal{I}) = 1$.

Strategy

- We try to produce a good assignment from the SDP output vectors.
- In particular, is it true that
  if $\mathrm{SDPOpt}(\mathcal{I}) = 1$ then $\mathcal{I}$ has a solution?
  This was suggested by Guruswami as the first step to attack the conjecture
- So, assume $\mathrm{SDPOpt}(\mathcal{I}) = 1$.
- It follows that $\mathbf{x}_a \mathbf{y}_b = 0$ for every $(a, b) \notin R_{xy}$

Strategy

- We try to produce a good assignment from the SDP output vectors.
- In particular, is it true that
  if $\mathrm{SDPOpt}(\mathcal{I}) = 1$ then $\mathcal{I}$ has a solution?
  This was suggested by Guruswami as the first step to attack the conjecture
- So, assume $\mathrm{SDPOpt}(\mathcal{I}) = 1$.
- It follows that $\mathbf{x}_a \mathbf{y}_b = 0$ for every $(a, b) \notin R_{xy}$
- Define $P_{xy} = \{(a, b) \in A^2 : \mathbf{x}_a \mathbf{y}_b > 0\}$.
  Replace R_{xy} with P_{xy}.
  If the new instance has a solution then the old one has a solution.

Strategy

- We try to produce a good assignment from the SDP output vectors.
- In particular, is it true that
  if $\mathrm{SDPOpt}(\mathcal{I}) = 1$ then $\mathcal{I}$ has a solution?
  This was suggested by Guruswami as the first step to attack the conjecture
- So, assume $\mathrm{SDPOpt}(\mathcal{I}) = 1$.
- It follows that $\mathbf{x}_a \mathbf{y}_b = 0$ for every $(a, b) \notin R_{xy}$
- Define $P_{xy} = \{(a, b) \in A^2 : \mathbf{x}_a \mathbf{y}_b > 0\}$.
  Replace R_{xy} with P_{xy}.
  If the new instance has a solution then the old one has a solution.
- Define $P_x = \{a \in A : \mathbf{x}_a \neq \mathbf{o}\}$.

Random facts about P_x, P_{xy}

$$P_{xy} = \{(a,b) \in A^2 : \mathbf{x}_a \mathbf{y}_b > 0\}, \quad P_x = \{a \in A : \mathbf{x}_a \neq \mathbf{o}\}$$

- P_{xy} is a subdirect subset of $P_x \times P_y$ (1-minimality)

Random facts about P_x, P_{xy}

$$P_{xy} = \{(a,b) \in A^2 : \mathbf{x}_a \mathbf{y}_b > 0\}, \quad P_x = \{a \in A : \mathbf{x}_a \neq \mathbf{o}\}$$

- P_{xy} is a subdirect subset of $P_x \times P_y$ (1-minimality)
  - It is a subset: If $\mathbf{x}_a \mathbf{y}_b > 0$ then $\mathbf{x}_a, \mathbf{y}_b \neq \mathbf{o}$
  - It is subdirect: If $\mathbf{x}_a \neq \mathbf{o}$ then $0 \neq \|\mathbf{x}_a\|^2 = \mathbf{x}_a \mathbf{y}_A$, therefore $\mathbf{x}_a \mathbf{y}_b \neq 0$ for some b

Random facts about P_x, P_{xy}

$$P_{xy} = \{(a,b) \in A^2 : \mathbf{x}_a \mathbf{y}_b > 0\}, \qquad P_x = \{a \in A : \mathbf{x}_a \neq \mathbf{o}\}$$

- P_{xy} is a subdirect subset of $P_x \times P_y$ (1-minimality)

For $B \subseteq P_x$ let $B + (x,y) = \{c \in A : (\exists b \in B)\ (b,c) \in P_{xy}\}$

Random facts about P_x, P_{xy}

$$P_{xy} = \{(a,b) \in A^2 : \mathbf{x}_a \mathbf{y}_b > 0\}, \qquad P_x = \{a \in A : \mathbf{x}_a \neq \mathbf{o}\}$$

- P_{xy} is a subdirect subset of $P_x \times P_y$ (1-minimality)

For $B \subseteq P_x$ let $B + (x,y) = \{c \in A : (\exists b \in B)\ (b,c) \in P_{xy}\}$

- For $B \subseteq P_x$, we have $\mathbf{y}_{B+(x,y)} = \mathbf{x}_B + \mathbf{w}$,
  where $\mathbf{w}\mathbf{x}_B = 0$, and $\mathbf{w} = \mathbf{o}$ iff $B = B + (x,y) - (x,y)$.

Random facts about P_x, P_{xy}

$$P_{xy} = \{(a,b) \in A^2 : \mathbf{x}_a \mathbf{y}_b > 0\}, \qquad P_x = \{a \in A : \mathbf{x}_a \neq \mathbf{o}\}$$

- P_{xy} is a subdirect subset of $P_x \times P_y$ (1-minimality)

For $B \subseteq P_x$ let $B + (x,y) = \{c \in A : (\exists b \in B)\ (b,c) \in P_{xy}\}$

- For $B \subseteq P_x$, we have $\mathbf{y}_{B+(x,y)} = \mathbf{x}_B + \mathbf{w}$,
  where $\mathbf{w}\mathbf{x}_B = 0$, and $\mathbf{w} = \mathbf{o}$ iff $B = B + (x,y) - (x,y)$.
  - $\mathbf{w}\mathbf{x}_B = (\mathbf{y}_{B+(x,y)} - \mathbf{x}_B)\mathbf{x}_B = \mathbf{y}_{B+(x,y)}\mathbf{x}_B - \mathbf{x}_B\mathbf{x}_B = \mathbf{y}_{B+(x,y)}\mathbf{x}_B - \mathbf{y}_A\mathbf{x}_B = -(\mathbf{y}_A - \mathbf{y}_{B+(x,y)})\mathbf{x}_B = -\mathbf{y}_{A-(B+(x,y))}\mathbf{x}_B = 0$
  - $\mathbf{w}\mathbf{w} = \cdots = \mathbf{x}_{A-B}\mathbf{y}_{B+(x,y)}$

Random facts about P_x, P_{xy}

$$P_{xy} = \{(a,b) \in A^2 : \mathbf{x}_a \mathbf{y}_b > 0\}, \qquad P_x = \{a \in A : \mathbf{x}_a \neq \mathbf{o}\}$$

- P_{xy} is a subdirect subset of $P_x \times P_y$ (1-minimality)

For $B \subseteq P_x$ let $B + (x,y) = \{c \in A : (\exists b \in B)\ (b,c) \in P_{xy}\}$

- For $B \subseteq P_x$, we have $\mathbf{y}_{B+(x,y)} = \mathbf{x}_B + \mathbf{w}$,
  where $\mathbf{w}\mathbf{x}_B = 0$, and $\mathbf{w} = \mathbf{o}$ iff $B = B + (x,y) - (x,y)$.

A (correct) sequence of variables is called a pattern
$B + p, B - p$ defined in a natural way for a pattern p

Random facts about P_x, P_{xy}

$$P_{xy} = \{(a,b) \in A^2 : \mathbf{x}_a \mathbf{y}_b > 0\}, \qquad P_x = \{a \in A : \mathbf{x}_a \neq \mathbf{o}\}$$

- P_{xy} is a subdirect subset of $P_x \times P_y$ (1-minimality)

For $B \subseteq P_x$ let $B + (x,y) = \{c \in A : (\exists b \in B)\ (b,c) \in P_{xy}\}$

- For $B \subseteq P_x$, we have $\mathbf{y}_{B+(x,y)} = \mathbf{x}_B + \mathbf{w}$,
  where $\mathbf{w}\mathbf{x}_B = 0$, and $\mathbf{w} = \mathbf{o}$ iff $B = B + (x,y) - (x,y)$.

A (correct) sequence of variables is called a pattern
$B + p, B - p$ defined in a natural way for a pattern p

For any $B \subseteq P_x$ and patterns p, q from x to x we have

- If $B + p = B$ then $B - p = B$

Random facts about P_x, P_{xy}

$$P_{xy} = \{(a,b) \in A^2 : \mathbf{x}_a \mathbf{y}_b > 0\}, \quad P_x = \{a \in A : \mathbf{x}_a \neq \mathbf{o}\}$$

- P_{xy} is a subdirect subset of $P_x \times P_y$ (1-minimality)

For $B \subseteq P_x$ let $B + (x,y) = \{c \in A : (\exists b \in B)\ (b,c) \in P_{xy}\}$

- For $B \subseteq P_x$, we have $\mathbf{y}_{B+(x,y)} = \mathbf{x}_B + \mathbf{w}$,
  where $\mathbf{w}\mathbf{x}_B = 0$, and $\mathbf{w} = \mathbf{o}$ iff $B = B + (x,y) - (x,y)$.

A (correct) sequence of variables is called a pattern
$B + p, B - p$ defined in a natural way for a pattern p

For any $B \subseteq P_x$ and patterns p, q from x to x we have

- If $B + p = B$ then $B - p = B$
- If $B + p + q = B$ then $B + p = B$

Random facts about P_x, P_{xy} - summary

The new instance with constraints $P_{xy}(x, y)$ and subsets $P_x \subseteq A, x \in V$ satisfies
(for every $x, y \in V$, $B \subseteq P_x$ and patterns p, q from x to x)

(P1) It is 1-minimal (P_{xy} is a subdirect subset of $P_x \times P_y$)

(P2) If $B + p = B$ then $B - p = B$

(P3) If $B + p + q = B$ then $B + p = B$

Weak Prague instance

Definition

An instance with constraints $P_{xy}(x, y)$ and subsets $P_x \subseteq A, x \in V$ is a weak Prague instance if
(for every $x, y \in V$, $B \subseteq P_x$ and patterns p, q from x to x)

(P1) It is 1-minimal (P_{xy} is a subdirect subset of $P_x \times P_y$)

(P2) If $B + p = B$ then $B - p = B$

(P3) If $B + p + q = B$ then $B + p = B$

Weak Prague instance

Definition

An instance with constraints $P_{xy}(x, y)$ and subsets $P_x \subseteq A, x \in V$ is a weak Prague instance if
(for every $x, y \in V$, $B \subseteq P_x$ and patterns p, q from x to x)

(P1) It is 1-minimal (P_{xy} is a subdirect subset of $P_x \times P_y$)

(P2) If $B + p = B$ then $B - p = B$

(P3) If $B + p + q = B$ then $B + p = B$

- Slightly weaker notion than Prague strategy
- Every Prague strategy has a solution (if P_{xy}'s are invariant under $\mathrm{Pol}\,\Gamma$...) BK

Weak Prague instance

Definition

An instance with constraints $P_{xy}(x, y)$ and subsets $P_x \subseteq A, x \in V$ is a weak Prague instance if
(for every $x, y \in V$, $B \subseteq P_x$ and patterns p, q from x to x)

(P1) It is 1-minimal (P_{xy} is a subdirect subset of $P_x \times P_y$)

(P2) If $B + p = B$ then $B - p = B$

(P3) If $B + p + q = B$ then $B + p = B$

- Slightly weaker notion than Prague strategy
- Every Prague strategy has a solution (if P_{xy}'s are invariant under $\mathrm{Pol}\,\Gamma$...) BK
- Every weak Prague strategy has a solution BK

General case

- $\mathrm{SDPOpt}(\Gamma) = 1 - \varepsilon$, ε small

General case

- $\mathrm{SDPOpt}(\Gamma) = 1 - \varepsilon$, ε small
- Choose δ (randomly with some distribution)
- Put $P_{xy} = \{(a, b) : \mathbf{x}_a \mathbf{x}_b > \delta\}$
- Put $P_x = \{a : ||\mathbf{x}_a||^2 > \delta\}$

General case

- $\mathrm{SDPOpt}(\Gamma) = 1 - \varepsilon$, ε small
- Choose δ (randomly with some distribution)
- Put $P_{xy} = \{(a, b) : \mathbf{x}_a \mathbf{x}_b > \delta\}$
- Put $P_x = \{a : ||\mathbf{x}_a||^2 > \delta\}$
- If δ not too tiny
  then for almost all x, y we have $P_{xy} \subseteq R_{xy}$.

General case

- $\mathrm{SDPOpt}(\Gamma) = 1 - \varepsilon$, ε small
- Choose δ (randomly with some distribution)
- Put $P_{xy} = \{(a, b) : \mathbf{x}_a \mathbf{x}_b > \delta\}$
- Put $P_x = \{a : ||\mathbf{x}_a||^2 > \delta\}$
- If δ not too tiny
  then for almost all x, y we have $P_{xy} \subseteq R_{xy}$.
- Give up other constraints

General case

- $\mathrm{SDPOpt}(\Gamma) = 1 - \varepsilon$, ε small
- Choose δ (randomly with some distribution)
- Put $P_{xy} = \{(a, b) : \mathbf{x}_a \mathbf{x}_b > \delta\}$
- Put $P_x = \{a : ||\mathbf{x}_a||^2 > \delta\}$
- If δ not too tiny
  then for almost all x, y we have $P_{xy} \subseteq R_{xy}$.
- Give up other constraints
- Now we can work with P_{xy} instead of R_{xy}

Enforcing (P1)

$$P_{xy} = \{(a,b) \in A^2 : \mathbf{x}_a \mathbf{y}_b > \delta\}, \quad P_x = \{a \in A : ||\mathbf{x}_a||^2 > \delta\}$$

we want (P1) P_{xy} is a subdirect subset of $P_x \times P_y$

Enforcing (P1)

$$P_{xy} = \{(a,b) \in A^2 : \mathbf{x}_a\mathbf{y}_b > \delta\}, \quad P_x = \{a \in A : ||\mathbf{x}_a||^2 > \delta\}$$

we want (P1) P_{xy} is a subdirect subset of $P_x \times P_y$

- Give up P_{xy} for which some $\mathbf{x}_a\mathbf{y}_b$ is in $(\delta - \mathit{enough}, \delta)$
  δ is chosen so that we don't delete too much

Enforcing (P1)

$$P_{xy} = \{(a,b) \in A^2 : \mathbf{x}_a \mathbf{y}_b > \delta\}, \quad P_x = \{a \in A : ||\mathbf{x}_a||^2 > \delta\}$$

we want (P1) P_{xy} is a subdirect subset of $P_x \times P_y$

- Give up P_{xy} for which some $\mathbf{x}_a \mathbf{y}_b$ is in $(\delta - \mathit{enough}, \delta)$
  δ is chosen so that we don't delete too much
- **(P1)** $P_{xy} \subseteq P_x \times P_y$**:**
  If $\mathbf{x}_a \mathbf{y}_b > \delta$ then $||\mathbf{x}_a||^2 = \mathbf{x}_a \mathbf{y}_B \geq \mathbf{x}_a \mathbf{y}_b > \delta$

Enforcing (P1)

$$P_{xy} = \{(a,b) \in A^2 : \mathbf{x}_a\mathbf{y}_b > \delta\}, \quad P_x = \{a \in A : ||\mathbf{x}_a||^2 > \delta\}$$

we want (P1) P_{xy} is a subdirect subset of $P_x \times P_y$

- Give up P_{xy} for which some $\mathbf{x}_a\mathbf{y}_b$ is in $(\delta - \mathit{enough}, \delta)$
  δ is chosen so that we don't delete too much
- **(P1)** $P_{xy} \subseteq P_x \times P_y$**:**
  If $\mathbf{x}_a\mathbf{y}_b > \delta$ then $||\mathbf{x}_a||^2 = \mathbf{x}_a\mathbf{y}_B \geq \mathbf{x}_a\mathbf{y}_b > \delta$
- **(P1) Subdirectness:** If $||\mathbf{x}_a||^2 = \mathbf{x}_a\mathbf{y}_B > \delta$ then $\mathbf{x}_a\mathbf{y}_b \geq \delta/|A|$. Then $\mathbf{x}_a\mathbf{y}_B \geq \delta$ (as $\delta/|A| > \delta - \mathit{enough}$)

Enforcing (P1)

$$P_{xy} = \{(a,b) \in A^2 : \mathbf{x}_a\mathbf{y}_b > \delta\}, \quad P_x = \{a \in A : ||\mathbf{x}_a||^2 > \delta\}$$

we want (P1) P_{xy} is a subdirect subset of $P_x \times P_y$

- Give up P_{xy} for which some $\mathbf{x}_a\mathbf{y}_b$ is in $(\delta - \mathit{enough}, \delta)$
  δ is chosen so that we don't delete too much
- **(P1)** $P_{xy} \subseteq P_x \times P_y$**:**
  If $\mathbf{x}_a\mathbf{y}_b > \delta$ then $||\mathbf{x}_a||^2 = \mathbf{x}_a\mathbf{y}_B \geq \mathbf{x}_a\mathbf{y}_b > \delta$
- **(P1) Subdirectness:** If $||\mathbf{x}_a||^2 = \mathbf{x}_a\mathbf{y}_B > \delta$ then
  $\mathbf{x}_a\mathbf{y}_b \geq \delta/|A|$. Then $\mathbf{x}_a\mathbf{y}_B \geq \delta$ (as $\delta/|A| > \delta - \mathit{enough}$)

Important property: $\mathbf{y}_{B+(x,y)}$ is

Enforcing (P1)

$$P_{xy} = \{(a,b) \in A^2 : \mathbf{x}_a \mathbf{y}_b > \delta\}, \quad P_x = \{a \in A : ||\mathbf{x}_a||^2 > \delta\}$$

we want (P1) P_{xy} is a subdirect subset of $P_x \times P_y$

- Give up P_{xy} for which some $\mathbf{x}_a \mathbf{y}_b$ is in $(\delta - \mathit{enough}, \delta)$
  δ is chosen so that we don't delete too much
- **(P1)** $P_{xy} \subseteq P_x \times P_y$**:**
  If $\mathbf{x}_a \mathbf{y}_b > \delta$ then $||\mathbf{x}_a||^2 = \mathbf{x}_a \mathbf{y}_B \geq \mathbf{x}_a \mathbf{y}_b > \delta$
- **(P1) Subdirectness:** If $||\mathbf{x}_a||^2 = \mathbf{x}_a \mathbf{y}_B > \delta$ then
  $\mathbf{x}_a \mathbf{y}_b \geq \delta / |A|$. Then $\mathbf{x}_a \mathbf{y}_B \geq \delta$ (as $\delta/|A| > \delta - \mathit{enough}$)

Important property: $\mathbf{y}_{B+(x,y)}$ is

- either almost ($\ll \delta$) the same as $\mathbf{x}_B$
  (in case that $B + (x,y,x) = B$),

Enforcing (P1)

$$P_{xy} = \{(a,b) \in A^2 : \mathbf{x}_a\mathbf{y}_b > \delta\}, \quad P_x = \{a \in A : ||\mathbf{x}_a||^2 > \delta\}$$

we want (P1) P_{xy} is a subdirect subset of $P_x \times P_y$

- Give up P_{xy} for which some $\mathbf{x}_a\mathbf{y}_b$ is in $(\delta - \mathit{enough}, \delta)$
  δ is chosen so that we don't delete too much
- **(P1)** $P_{xy} \subseteq P_x \times P_y$**:**
  If $\mathbf{x}_a\mathbf{y}_b > \delta$ then $||\mathbf{x}_a||^2 = \mathbf{x}_a\mathbf{y}_B \geq \mathbf{x}_a\mathbf{y}_b > \delta$
- **(P1) Subdirectness:** If $||\mathbf{x}_a||^2 = \mathbf{x}_a\mathbf{y}_B > \delta$ then
  $\mathbf{x}_a\mathbf{y}_b \geq \delta/|A|$. Then $\mathbf{x}_a\mathbf{y}_B \geq \delta$ (as $\delta/|A| > \delta - \mathit{enough}$)

Important property: $\mathbf{y}_{B+(x,y)}$ is

- either almost ($\ll \delta$) the same as $\mathbf{x}_B$
  (in case that $B + (x,y,x) = B$),
- or significantly ($> \delta$) longer than $\mathbf{x}_B$ (otherwise)

Enforcing (P2)

$$P_{xy} = \{(a,b) \in A^2 : \mathbf{x}_a \mathbf{y}_b > \delta\}, \quad P_x = \{a \in A : ||\mathbf{x}_a||^2 > \delta\}$$

We want (P2) If $B + p = B$ then $B - p = B$

Enforcing (P2)

$$P_{xy} = \{(a,b) \in A^2 : \mathbf{x}_a \mathbf{y}_b > \delta\}, \quad P_x = \{a \in A : ||\mathbf{x}_a||^2 > \delta\}$$

We want (P2) If $B + p = B$ then $B - p = B$

- Divide the unit ball into layers
  (thickness about δ, randomly shifted)

Enforcing (P2)

$$P_{xy} = \{(a,b) \in A^2 : \mathbf{x}_a \mathbf{y}_b > \delta\}, \quad P_x = \{a \in A : ||\mathbf{x}_a||^2 > \delta\}$$

We want (P2) If $B + p = B$ then $B - p = B$

- Divide the unit ball into layers
  (thickness about δ, randomly shifted)
- Give up P_{xy} if there are almost the same vectors $\mathbf{x}_B, \mathbf{y}_{B+(x,y)}$ in different layers.

Enforcing (P2)

$$P_{xy} = \{(a, b) \in A^2 : \mathbf{x}_a \mathbf{y}_b > \delta\}, \quad P_x = \{a \in A : ||\mathbf{x}_a||^2 > \delta\}$$

We want (P2) If $B + p = B$ then $B - p = B$

- Divide the unit ball into layers
  (thickness about δ, randomly shifted)
- Give up P_{xy} if there are almost the same vectors $\mathbf{x}_B, \mathbf{y}_{B+(x,y)}$ in different layers.

Recall that $\mathbf{y}_{B+(x,y)}$ is

- either almost the same as $\mathbf{x}_B$ (in case that $B + (x, y, x) = B$),
- or significantly longer ($> \delta$) than $\mathbf{x}_B \Rightarrow$ vector jumps to higher layer

This guarantees (P2)

Enforcing (P2)

$$P_{xy} = \{(a,b) \in A^2 : \mathbf{x}_a \mathbf{y}_b > \delta\}, \quad P_x = \{a \in A : ||\mathbf{x}_a||^2 > \delta\}$$

We want (P2) If $B + p = B$ then $B - p = B$

- Divide the unit ball into layers
  (thickness about δ, randomly shifted)
- Give up P_{xy} if there are almost the same vectors $\mathbf{x}_B, \mathbf{y}_{B+(x,y)}$ in different layers.

Recall that $\mathbf{y}_{B+(x,y)}$ is

- either almost the same as $\mathbf{x}_B$ (in case that $B + (x,y,x) = B$),
- or significantly longer ($> \delta$) than $\mathbf{x}_B \Rightarrow$ vector jumps to higher layer

This guarantees (P2)

(note: so far we only used lengths $\Rightarrow$ can be done for LP relaxation)

Enforcing (P3)

$$P_{xy} = \{(a, b) \in A^2 : \mathbf{x}_a \mathbf{y}_b > \delta\}, \quad P_x = \{a \in A : ||\mathbf{x}_a||^2 > \delta\}$$

(P3) If $B + p + q = B$ then $B + p = B$

Enforcing (P3)

$P_{xy} = \{(a,b) \in A^2 : \mathbf{x}_a \mathbf{y}_b > \delta\}, \quad P_x = \{a \in A : ||\mathbf{x}_a||^2 > \delta\}$

(P3) If $B + p + q = B$ then $B + p = B$

- Choose sufficiently many hyperplanes (randomly)

Enforcing (P3)

$P_{xy} = \{(a,b) \in A^2 : \mathbf{x}_a \mathbf{y}_b > \delta\}, \quad P_x = \{a \in A : ||\mathbf{x}_a||^2 > \delta\}$

(P3) If $B + p + q = B$ then $B + p = B$

- Choose sufficiently many hyperplanes (randomly)

(x) Give up variables x for which some pair $\mathbf{x}_B, \mathbf{x}_C$ is not cut

Enforcing (P3)

$$P_{xy} = \{(a,b) \in A^2 : \mathbf{x}_a \mathbf{y}_b > \delta\}, \quad P_x = \{a \in A : ||\mathbf{x}_a||^2 > \delta\}$$

(P3) If $B + p + q = B$ then $B + p = B$

- Choose sufficiently many hyperplanes (randomly)

(x) Give up variables x for which some pair $\mathbf{x}_B, \mathbf{x}_C$ is not cut

(y) Give up constraints P_{xy} for which there are almost the same vectors $\mathbf{x}_B, \mathbf{y}_{B+(x,y)}$ which are cut

Enforcing (P3)

$$P_{xy} = \{(a,b) \in A^2 : \mathbf{x}_a \mathbf{y}_b > \delta\}, \quad P_x = \{a \in A : ||\mathbf{x}_a||^2 > \delta\}$$

(P3) If $B + p + q = B$ then $B + p = B$

- Choose sufficiently many hyperplanes (randomly)

(x) Give up variables x for which some pair $\mathbf{x}_B, \mathbf{x}_C$ is not cut

(y) Give up constraints P_{xy} for which there are almost the same vectors $\mathbf{x}_B, \mathbf{y}_{B+(x,y)}$ which are cut

- This guarantees (P3)

Enforcing (P3)

$$P_{xy} = \{(a,b) \in A^2 : \mathbf{x}_a \mathbf{y}_b > \delta\}, \quad P_x = \{a \in A : ||\mathbf{x}_a||^2 > \delta\}$$

(P3) If $B + p + q = B$ then $B + p = B$

- Choose sufficiently many hyperplanes (randomly)

(x) Give up variables x for which some pair $\mathbf{x}_B, \mathbf{x}_C$ is not cut

(y) Give up constraints P_{xy} for which there are almost the same vectors $\mathbf{x}_B, \mathbf{y}_{B+(x,y)}$ which are cut

- This guarantees (P3)
- Remark: Different number of hyperplanes is used for different layers otherwise (x) or (y) would delete too much

Enforcing (P3)

$$P_{xy} = \{(a,b) \in A^2 : \mathbf{x}_a \mathbf{y}_b > \delta\}, \quad P_x = \{a \in A : ||\mathbf{x}_a||^2 > \delta\}$$

(P3) If $B + p + q = B$ then $B + p = B$

- Choose sufficiently many hyperplanes (randomly)

(x) Give up variables x for which some pair $\mathbf{x}_B, \mathbf{x}_C$ is not cut

(y) Give up constraints P_{xy} for which there are almost the same vectors $\mathbf{x}_B, \mathbf{y}_{B+(x,y)}$ which are cut

- This guarantees (P3)
- Remark: Different number of hyperplanes is used for different layers otherwise (x) or (y) would delete too much

Now we have a Prague instance. Algebraic closure has a solution

Final remarks

- Is the quantitative dependence optimal?

Final remarks

- Is the quantitative dependence optimal?
- How to improve derandomization to match the randomized version?

Final remarks

- Is the quantitative dependence optimal?
- How to improve derandomization to match the randomized version?
- What can we say about the quantitative dependence on ε in general?

Final remarks

- Is the quantitative dependence optimal?
- How to improve derandomization to match the randomized version?
- What can we say about the quantitative dependence on ε in general?
  - Guess: NU $\Rightarrow$ polynomial loss, or SD($\wedge$) $\Rightarrow$ polynomial loss

Final remarks

- Is the quantitative dependence optimal?
- How to improve derandomization to match the randomized version?
- What can we say about the quantitative dependence on ε in general?
  - Guess: NU $\Rightarrow$ polynomial loss, or SD($\wedge$) $\Rightarrow$ polynomial loss
  - Candidates for hardness: 2-semilattices?

Final remarks

- Is the quantitative dependence optimal?
- How to improve derandomization to match the randomized version?
- What can we say about the quantitative dependence on ε in general?
  - Guess: NU $\Rightarrow$ polynomial loss, or SD($\wedge$) $\Rightarrow$ polynomial loss
  - Candidates for hardness: 2-semilattices?
- Explore further SDP,LP $\leftrightarrow$ consistency notions

Final remarks

- Is the quantitative dependence optimal?
- How to improve derandomization to match the randomized version?
- What can we say about the quantitative dependence on ε in general?
  - Guess: NU $\Rightarrow$ polynomial loss, or SD($\wedge$) $\Rightarrow$ polynomial loss
  - Candidates for hardness: 2-semilattices?
- Explore further SDP,LP $\leftrightarrow$ consistency notions
- Thank you!