Optimal Sparsification for Some Binary CSPs Using Low-Degree Polynomials

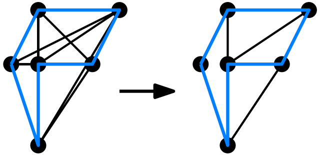
Bart M. P. Jansen and Astrid Pieterse

August 23, 2016

Technische Universiteit Eindhoven University of Technology

Sparsification

- Polynomial-time preprocessing
- Making a graph or logical structure less “dense”
- Graph problems
  - Reduce the number of edges
- Satisfiability
  - Reducing the number of clauses
  - Keeping the yes/no-answer



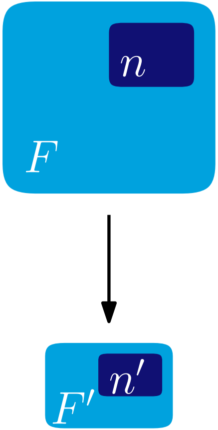
$(x \vee y) \wedge (\quad \vee \ldots \vee \quad) \wedge \ldots$

Sparsification

Reduce the size of an input instance, before solving the problem.

Sparsification of satisfaction problems

- Algorithm mapping formula F on n variables to F'
  - The running time is polynomial
  - $|F'|, n'$ are bounded by $f(n)$
  - F' is a YES-instance if and only if F is a YES-instance
- $f(n)$ is the *size*



Previous results

No polynomial-time sparsification maintaining the solution (unless $\mathrm{NP} \subseteq \mathrm{coNP}/\mathrm{poly}$)

- d-CNF-SAT to $O(n^{d-\varepsilon})$
  - Dell, van Melkebeek (J ACM14, STOC10)
- TREEWIDTH to $O(n^{2-\varepsilon})$
  - Jansen (Algorithmica 15, IPEC13)
- HAMILTONIAN CYCLE to $O(n^{2-\varepsilon})$
  - And a number of other graph problems
  - Jansen, Pieterse (IPEC15)

But d-NAE-SAT has a sparsification of size $\widetilde{O}(n^{d-1})$!

- Jansen, Pieterse (IPEC15)

Another variation: EXACT SATISFIABILITY

Previous results

No polynomial-time sparsification maintaining the solution (unless $\mathrm{NP} \subseteq \mathrm{coNP/poly}$)

- d-CNF-SAT to $O(n^{d-\varepsilon})$
  - Dell, van Melkebeek (J ACM14, STOC10)
- TREEWIDTH to $O(n^{2-\varepsilon})$
  - Jansen (Algorithmica 15, IPEC13)
- HAMILTONIAN CYCLE to $O(n^{2-\varepsilon})$
  - And a number of other graph problems
  - Jansen, Pieterse (IPEC15)

But d-NAE-SAT has a sparsification of size $\widetilde{O}(n^{d-1})$!

- Jansen, Pieterse (IPEC15)

Another variation: EXACT SATISFIABILITY

Previous results

No polynomial-time sparsification maintaining the solution (unless $\mathrm{NP} \subseteq \mathrm{coNP/poly}$)

- d-CNF-SAT to $O(n^{d-\varepsilon})$
  - Dell, van Melkebeek (J ACM14, STOC10)
- TREEWIDTH to $O(n^{2-\varepsilon})$
  - Jansen (Algorithmica 15, IPEC13)
- HAMILTONIAN CYCLE to $O(n^{2-\varepsilon})$
  - And a number of other graph problems
  - Jansen, Pieterse (IPEC15)

But d-NAE-SAT has a sparsification of size $\widetilde{O}(n^{d-1})$!

- Jansen, Pieterse (IPEC15)

Another variation: EXACT SATISFIABILITY

Exact Satisfiability

Input A formula in CNF form, consisting of clauses, each consisting of a number of literals.

$$\underbrace{\{\neg x, \neg y\}}_{\text{clause}} \wedge \{\neg y, z\} \wedge \{x, z\}$$

Parameter The number of variables n.

Question Does there exists an assignment to the variables, such that each clause contains exactly one *true* literal?

Sparsification for Exact Satisfiability

Example

Let $x, y, z \in \{0, 1\}$ (where 0 is false, 1 is true), then

$\{\neg x, \neg y\} \wedge \{\neg y, z\} \wedge \{x, z\}$ is exact-satisfiable

$$\Leftrightarrow \begin{aligned} (1-x)+(1-y) &= 1 \\ (1-y)+z &= 1 \\ x+z &= 1 \end{aligned} \quad \Leftrightarrow \quad \begin{aligned} x+y &= 1 \\ z-y &= 0 \\ x+z &= 1 \end{aligned}$$

- Clause $\{x, z\}$ is satisfied if the other two clauses are satisfied.

Sparsification for Exact Satisfiability

Example

Let $x, y, z \in \{0, 1\}$ (where 0 is false, 1 is true), then

$\{\neg x, \neg y\} \wedge \{\neg y, z\} \wedge \{x, z\}$ is exact-satisfiable

$$\Leftrightarrow \begin{aligned} (1 - x) + (1 - y) &= 1 \\ (1 - y) + z &= 1 \\ x + z &= 1 \end{aligned} \quad \Leftrightarrow \quad \begin{aligned} x + y &= 1 \\ z - y &= 0 \\ x + z &= 1 \end{aligned}$$

- Clause $\{x, z\}$ is satisfied if the other two clauses are satisfied.

Sparsification for Exact Satisfiability

Example

Let $x, y, z \in \{0, 1\}$ (where 0 is false, 1 is true), then

$$\begin{array}{lcl} \{\neg x, \neg y\} \wedge \{\neg y, z\} \wedge \{x, z\} & & \begin{aligned} (1 - x) + (1 - y) &= 1 \\ (1 - y) + z &= 1 \\ x + z &= 1 \end{aligned} & & \begin{aligned} x + y &= 1 \\ z - y &= 0 \\ x + z &= 1 \end{aligned} \\ \text{is exact-satisfiable} & \Leftrightarrow & & \Leftrightarrow & \end{array}$$

- Clause $\{x, z\}$ is satisfied if the other two clauses are satisfied.

Sparsification for Exact Satisfiability

Example

Let $x, y, z \in \{0, 1\}$ (where 0 is false, 1 is true), then

$$\begin{array}{l} \{\neg x, \neg y\} \wedge \{\neg y, z\} \wedge \{x, z\} \\ \text{is exact-satisfiable} \end{array} \Leftrightarrow \begin{array}{lr} (1-x)+(1-y) & =1 \\ (1-y)+z & =1 \\ x+z & =1 \end{array} \Leftrightarrow \begin{array}{rl} x+y & =1 \\ z-y & =0 \\ \hline x+z & =1 \end{array}+$$

- Clause $\{x, z\}$ is satisfied if the other two clauses are satisfied.

Sparsification for Exact Satisfiability

Example

Let $x, y, z \in \{0, 1\}$ (where 0 is false, 1 is true), then

$\{\neg x, \neg y\} \wedge \{\neg y, z\} \wedge \{x, z\}$ is exact-satisfiable

$$\Leftrightarrow \begin{aligned} (1 - x) + (1 - y) &= 1 \\ (1 - y) + z &= 1 \\ x + z &= 1 \end{aligned} \quad \Leftrightarrow \quad \begin{array}{l} x + y = 1 \\ \underline{z - y = 0}\,+ \\ x + z = 1 \end{array}$$

- Clause $\{x, z\}$ is satisfied if the other two clauses are satisfied.

Sparsification for Exact Satisfiability

Algorithm

- Given formula F
- Rewrite by giving an linear equation for each clause
- Find a basis of the row-space
  - Use Gaussian elimination
- Remove constraints not in the basis

$$\begin{pmatrix} 1 & 0 & \dots & 1 \\ 1 & 1 & \dots & 0 \\ \dots & \dots & \dots & \dots \\ \dots & \dots & \dots & \dots \\ 0 & -1 & \dots & 1 \end{pmatrix} \begin{pmatrix} x_1 \\ \vdots \\ x_n \end{pmatrix} = \begin{pmatrix} 1 \\ 1 \\ \vdots \\ \vdots \\ 0 \end{pmatrix}$$

Sparsification for Exact Satisfiability

Algorithm

- Given formula F
- Rewrite by giving an linear equation for each clause
- Find a basis of the row-space
  - Use Gaussian elimination
- Remove constraints not in the basis

$$\begin{pmatrix} 1 & 0 & \dots & 1 \\ 1 & 1 & \dots & 0 \\ \dots & \dots & \dots & \dots \\ \dots & \dots & \dots & \dots \\ 0 & -1 & \dots & 1 \end{pmatrix} \begin{pmatrix} x_1 \\ \vdots \\ x_n \end{pmatrix} = \begin{pmatrix} 1 \\ 1 \\ \vdots \\ \vdots \\ 0 \end{pmatrix}$$

Sparsification for Exact Satisfiability

Algorithm

- Given formula F
- Rewrite by giving an linear equation for each clause
- Find a basis of the row-space
  - Use Gaussian elimination
- Remove constraints not in the basis

$$\left(\begin{array}{cccc|c} 1 & 0 & \dots & 1 & 1 \\ 1 & 1 & \dots & 0 & 1 \\ \dots & \dots & \dots & \dots & \\ \dots & \dots & \dots & \dots & \\ 0 & -1 & \dots & 1 & 0 \end{array}\right)$$

Sparsification for Exact Satisfiability

Algorithm

- Given formula F
- Rewrite by giving an linear equation for each clause
- Find a basis of the row-space
  - Use Gaussian elimination
- Remove constraints not in the basis

$$\left(\begin{array}{cccc|c} 1 & 0 & \dots & 1 & 1 \\ 1 & 1 & \dots & 0 & 1 \\ \dots & \dots & \dots & \dots & \dots \\ \dots & \dots & \dots & \dots & \dots \\ 0 & -1 & \dots & 1 & 0 \end{array}\right)$$

Sparsification for Exact Satisfiability

Algorithm

- Given formula F
- Rewrite by giving an linear equation for each clause
- Find a basis of the row-space
  - Use Gaussian elimination
- Remove constraints not in the basis

$$\left(\begin{array}{cccc|c} 1 & 0 & \dots & 1 & 1 \\ 1 & 1 & \dots & 0 & 1 \\ \dots & \dots & \dots & \dots & \dots \end{array}\right)$$

Sparsification for Exact Satisfiability

Correctness

- Polynomial time
- yes-instance after removing constraints $\Rightarrow F$ is a yes-instance

Size

- Matrix size (#clauses) $\times (n+1)$
- At most $n+1$ clauses remaining
- For bounded clause $\widetilde{O}(n)$ bits, else $\widetilde{O}(n^2)$

Constraint Satisfaction Problems

Constraints over set of variables V

- We consider $0/1$-variables
- Constraint $R(x_1, \ldots, x_k)$ applies relation R to variables $x_1, \ldots, x_k \in V$

Schaefer's dichotomy theorem

- Depending on the properties of used relations R
  - Polynomial time solvable
  - or NP-hard

Can we get a similar classification for sparsification bounds?

Constraint Satisfaction Problems

Constraints over set of variables V

- We consider $0/1$-variables
- Constraint $R(x_1, \ldots, x_k)$ applies relation R to variables $x_1, \ldots, x_k \in V$

Schaefer's dichotomy theorem

- Depending on the properties of used relations R
  - Polynomial time solvable
  - or NP-hard

Can we get a similar classification for sparsification bounds?

Constraint Satisfaction Problems

Constraints over set of variables V

- We consider $0/1$-variables
- Constraint $R(x_1, \ldots, x_k)$ applies relation R to variables $x_1, \ldots, x_k \in V$

Schaefer's dichotomy theorem

- Depending on the properties of used relations R
  - Polynomial time solvable
  - or NP-hard

Can we get a similar classification for sparsification bounds?

Relating results so far

$\widetilde{O}(n^d)$	d-CNF-SAT	is d-OPTIONS SAT use $S := \{1, 2, \ldots, d\}$
$\widetilde{O}(n^{d-1})$	d-NAE-SAT	is $(d-1)$-OPTIONS SAT use $S := \{1, 2, \ldots, d-1\}$
$\widetilde{O}(n)$	d-EXACT-SAT	is 1-OPTIONS SAT use $S := \{1\}$

c-Options Sat

Input A set of clauses over variables V and set $S_i \subset \mathbb{N}$ with $|S_i| \leqslant c$.

Parameter The number of variables n.

Question Does there exists an assignment to the variables, such the number of *true* literals in clause i lies in S_i?

Relating results so far

$\widetilde{O}(n^d)$	d-CNF-SAT	is d-Options SAT use $S := \{1, 2, \ldots, d\}$
$\widetilde{O}(n^{d-1})$	d-NAE-SAT	is $(d-1)$-Options SAT use $S := \{1, 2, \ldots, d-1\}$
$\widetilde{O}(n)$	d-Exact-SAT	is 1-Options SAT use $S := \{1\}$

c-Options Sat

Input A set of clauses over variables V and set $S_i \subset \mathbb{N}$ with $|S_i| \leqslant c$.

Parameter The number of variables n.

Question Does there exists an assignment to the variables, such the number of *true* literals in clause i lies in S_i?

Relating results so far

$\widetilde{O}(n^d)$	d-CNF-SAT	is d-Options sat use $S := \{1, 2, \ldots, d\}$
$\widetilde{O}(n^{d-1})$	d-NAE-SAT	is $(d-1)$-Options sat use $S := \{1, 2, \ldots, d-1\}$
$\widetilde{O}(n)$	d-Exact-SAT	is 1-Options sat use $S := \{1\}$

c-Options Sat

Input A set of clauses over variables V and set $S_i \subset \mathbb{N}$ with $|S_i| \leqslant c$.

Parameter The number of variables n.

Question Does there exists an assignment to the variables, such the number of *true* literals in clause i lies in S_i?

c-Options Sat: Results

c-OPTIONS SAT has a sparsification with $O(n^c)$ clauses

- Bitsize $\widetilde{O}(n^{c+1})$ if no bound on clause size

c-Options Sat represented by polynomials

Given S_i we can find a polynomial f of degree c

$$f(x_1, \dots, x_k) = 0 \Leftrightarrow \text{clause } (x_1, \dots, x_k) \text{ is satisfied}$$

Example $S := \{1, 3\}$, $c = 2$, clause (w, x, y, z)

- Satisfied if $w + x + y + z = a \in \{1, 3\}$
- Let $G(a) = (a - 1) \cdot (a - 3)$
  - $G(1) = G(3) = 0$
- Let $f(w, x, y, z) := G(w + x + y + z)$

c-Options Sat represented by polynomials

Given S_i we can find a polynomial f of degree c

$$f(x_1, \ldots, x_k) = 0 \Leftrightarrow \text{clause } (x_1, \ldots, x_k) \text{ is satisfied}$$

Example $S := \{1, 3\}$, $c = 2$, clause (w, x, y, z)

- Satisfied if $w + x + y + z = a \in \{1, 3\}$
- Let $G(a) = (a - 1) \cdot (a - 3)$
  - $G(1) = G(3) = 0$
- Let $f(w, x, y, z) := G(w + x + y + z)$

c-Polynomial root CSP

Input	A list of polynomial equalities of the form $f(x_1, \ldots, x_k) = 0$ where f has degree at most c.
Parameter	The number of variables n.
Question	Does there exists an assignment to the variables, such that all equalities are satisfied?

c-Polynomial non-root CSP

Input	A list of polynomial inequalities $f(x_1, \ldots, x_n) \neq 0$ where f has degree at most c.
Parameter	The number of variables n.
Question	Does there exists an assignment to the variables, such that all inequalities are satisfied?

Our results

c-Polynomial -		over $\mathbb{R}$	$\mathbb{Z}$ mod p	$\mathbb{Z}$ mod m
root CSP	UB[1]	$\widetilde{O}(n^{c+1})$	$\widetilde{O}(n^{c+1})$	?
	LB[2]	$\Omega(n^{c+1-\varepsilon})$	$\Omega(n^{c+1-\varepsilon})$	$\Omega(n^{c+1-\varepsilon})$
non-root CSP	UB[1]	-	$\widetilde{O}(n^{c(p-1)+1})$	?
	LB[2]	Exp	$\Omega(n^{c(p-1)-\varepsilon})$	$\Omega(n^{d^r/2-\varepsilon})$

[1] Assuming clauses of size $\leqslant n$ are encoded in $\widetilde{O}(n)$ bits.
[2] $\forall \varepsilon > 0$, assuming $\mathrm{NP} \not\subseteq \mathrm{coNP/poly}$

A sparsification for c-Polynomial root CSP

Algorithm

- Given input F
- One matrix row for each polynomial equality
  - $f(x, y) = \alpha_0 + \alpha_1 x + \alpha_2 y + \alpha_3 xy + \ldots$
  - $g(x, y) = \beta_0 + \beta_1 x + \beta_2 y + \beta_3 xy + \ldots$

$$\begin{array}{c c} & \begin{array}{ccccc} 1 & x & y & xy & \ldots \end{array} \\ \begin{array}{c} f \\ g \\ \ldots \end{array} & \begin{pmatrix} \alpha_0 & \alpha_1 & \alpha_2 & \alpha_3 & \ldots \\ \beta_0 & \beta_1 & \beta_2 & \beta_3 & \ldots \\ \ldots & \ldots & \ldots & \ldots & \ldots \end{pmatrix} \end{array}$$

- Find redundant constraints
  - Do Gaussian elimination to find a basis of the row-space of the matrix
- Remove them

A sparsification for c-Polynomial root CSP

Correctness

Removed constraints are a linear combination of remaining constraints.

Size

- One column for each coefficient
  - $O(n^c)$ multilinear monomials gives $O(n^c)$ columns
- At most $O(n^c)$ remaining constraints

Generalizations

Works with polynomial equalities over any field

- For example, $\mathbb{Z}/p\mathbb{Z}$

A sparsification for c-Polynomial root CSP

Correctness

Removed constraints are a linear combination of remaining constraints.

Size

- One column for each coefficient
  - $O(n^c)$ multilinear monomials gives $O(n^c)$ columns
- At most $O(n^c)$ remaining constraints

Generalizations

Works with polynomial equalities over any field

- For example, $\mathbb{Z}/p\mathbb{Z}$

A sparsification for c-Polynomial root CSP

Correctness

Removed constraints are a linear combination of remaining constraints.

Size

- One column for each coefficient
  - $O(n^c)$ multilinear monomials gives $O(n^c)$ columns
- At most $O(n^c)$ remaining constraints

Generalizations

Works with polynomial equalities over any field

- For example, $\mathbb{Z}/p\mathbb{Z}$

Lower bound

Is the given sparsification "optimal"?

- $O(n^c)$ remaining constraints
- Sparsification size $\widetilde{O}(n^{c+1})$
- We showed that the problem has no sparsification of size $O(n^{c+1-\varepsilon})$, if $NP \not\subseteq coNP/poly$
  - "Simple" polynomials
  - Using cross-composition, details in the paper

c-Polynomial non-root CSP over $\mathbb{R}$

1-Polynomial non-root CSP can express CNF-SAT

- Clause $(x \vee y \vee z)$ is equivalent to

$$x + y + z \neq 0$$

d-CNF-SAT does not have a (more general type of) sparsification of size $O(n^{d-\varepsilon})$

- Unless $\mathrm{NP} \subseteq \mathrm{coNP/poly}$

No polynomial sparsification for 1-Polynomial non-root CSP

c-Polynomial non-root CSP mod p

Use the sparsification for c-Polynomial root CSP

- Consider $f(x_1, \ldots, x_k) \neq 0 \pmod{p}$
- Equivalent to $f(x_1, \ldots, x_k) \in \{1, 2, \ldots, p-1\} \pmod{p}$
- Let $G(x) := (x-1) \cdot (x-2) \cdots (x-p+1)$
- $f(x_1, \ldots, x_k) \neq 0 \pmod{p} \Leftrightarrow G(f(x_1, \ldots, x_k)) = 0 \pmod{p}$
  - has degree $c(p-1)$

Instance for $c(p-1)$-Polynomial root CSP, constraints replaced by $G \circ f$

- $O(n^{c(p-1)})$ constraints remain
- LB: no sparsification of size $O(n^{c(p-1)-\varepsilon})$

Strategy fails when p is not prime!

Lower bound: c-Polynomial non-root CSP mod m

Why does our strategy fail modulo a non-prime?

- Counter example for m not prime
- $m = 6$, $c = 3$, procedure would give $O(n^{c(m-1)}) = O(n^{15})$ clauses
- But, there is a degree-3 polynomial f such that

$$f(x_1, x_2, \ldots, x_{27}) \neq 0 \pmod{6} \Leftrightarrow (x_1 \vee x_2 \vee \ldots \vee x_{27})$$

- No sparsification of size $O(n^{27-\varepsilon})$ possible
  - Unless $\mathrm{NP} \subseteq \mathrm{coNP/poly}$

Conclusion

- Optimal sparsifications for two types of CSPs
- Relates existing results
- Open problems:
  - CSPs represented by polynomial (in)equalities over non-field
  - “the number of satisfied literals is one or two, modulo six”

Thank you!

Conclusion

- Optimal sparsifications for two types of CSPs
- Relates existing results
- Open problems:
  - CSPs represented by polynomial (in)equalities over non-field
  - “the number of satisfied literals is one or two, modulo six”

Thank you!