Testing assignments to constraint satisfaction problems

H. Chen[1] M. Valeriote[2] Y. Yoshida[3]

[1]University País Vasco

[2]McMaster University

[3]NII, Tokyo

{Symmetry, Logic, Computation}, 10 November 2016

The Constraint Satisfaction Problem

Definition

- Let **A** be a finite relational structure $\langle A, R_1, \ldots, R_k \rangle$ where for each i, $R_i \subseteq A^{ar(R_i)}$, with $ar(R_i) \in \mathbb{N}$ the arity of R_i. **A** will sometimes be referred to as a template.
- An instance of CSP(**A**) is a pair $I = (V, \mathcal{C})$ with
  - V a nonempty, finite set of variables,
  - $\mathcal{C}$ a finite set of constraints $\{C \colon C \in \mathcal{C}\}$ where each C is a pair $(\vec{s}, R)$ with
    - $R = R_i$ for some $1 \leq i \leq k$, called the constraint relation of C.
    - $\vec{s}$ a tuple of variables of length $ar(R_i)$, called the scope of C.
- A solution to I is a function (assignment) $f : V \to A$ such that for each $C = (\vec{s}, R) \in \mathcal{C}$, $f(\vec{s}) \in R$.

Graph colourability

Example (Graph k-colourability)

- Let $\mathbf{K}_k$ be the structure $\langle\{1,2,\ldots,k\},\neq_k\rangle$, where $\neq_k$ is the not-equals relation on $\{1,2,\ldots,k\}$.
- An instance of the graph k-colourability problem, i.e., a finite graph $\mathbf{G}=(V,E)$, can be viewed as the instance of CSP$(\mathbf{K}_k)$ with variable set V and constraint set

$$\{((v,w),\neq_k)\colon (v,w)\in E\}.$$

- The set of k-colourings of $\mathbf{G}$ is exactly the set of solutions of this instance.
- It is well known that the decision problem for graph k-colourability is in **P** for $k=2$ and is **NP**-complete for $k>2$.

The CSP decision problem

The Decision problem

For a template $\mathbf{A}$, the decision problem for CSP$(\mathbf{A})$ is:

Given an instance I of CSP$(\mathbf{A})$*, does I have a solution?*

Feder-Vardi Dichotomy Conjecture

For any template $\mathbf{A}$, the decision problem for CSP$(\mathbf{A})$ is either in **P** or is **NP**-complete.

Testing assignments to CSPs

Deciding an assignment

- Let **A** be a structure and $I = (V, \mathcal{C})$ an instance of CSP(**A**).
- Given a function $f\colon V \to A$, we can decide whether or not f is a solution to I in time linear in $|I|$.

Testing an assignment

- Question: Can we more quickly test if an assignment satisfies an instance of CSP(**A**) or is far from any satisfying assignment?
- Answer: In general, no, but for certain templates **A**, testing can be carried out in constant time (independent of $|V|$, after some pre-processing of the instance).

A distance function for assignments

Definition

Let

- $\mathbf{A}$ be a template,
- $I = (V, \mathcal{C})$ an instance of CSP($\mathbf{A}$) and
- $w\colon V \to [0,1]$ with $\sum_{v \in V} w(v) = 1$, a weight function.

For assignments f, $g\colon V \to A$ of I, the distance between f and g is:

$$\text{dist}(f,g) = \sum \{ w(v)\colon v \in V, f(v) \neq g(v) \}.$$

For $\varepsilon \in (0,1)$ we say that an assignment f is ε-far from satisfying I if $\text{dist}(f,g) > \varepsilon$ for all satisfying assignments g of I.

Testing assignments to CSPs

Definition (ε-tester)

Let **A** be a template. A tester for CSP(**A**) is an algorithm with the following input and output:

- Input:
  - $\varepsilon \in (0,1)$,
  - a (satisfiable) instance $I = (V, \mathcal{C})$ of CSP(**A**),
  - a weight function $w\colon V \to [0,1]$ with $\sum_{v \in V} w(v) = 1$, and
  - query access to an assignment $f\colon V \to A$.
- Output:
  - YES, with probability $\geq 2/3$ if f satisfies I.
  - NO, with probability $\geq 2/3$ if f is ε-far from satisfying I.

Remark

A tester is one-sided if it always outputs YES for satisfying assignments.

Query complexity

- We measure the efficiency of a tester by the number of queries it makes of the given assignment.
- The query complexity of a tester is constant/sublinear/linear if, for any ε, the number of queries it makes of the given assignment is constant/sublinear/linear in the number of variables of the assignment.
- The query complexity of CSP($\mathbf{A}$) is constant/sublinear/linear if it has a tester with that query complexity.

Remark

The query complexity of any CSP($\mathbf{A}$) is at worst linear, since we can devise a tester that queries all of the values of a given assignment.

Query complexity of CSP($\mathbf{A}$)

Problem

For a given template $\mathbf{A}$, determine the query complexity of CSP($\mathbf{A}$).

Some known results

CSP	Query complexity
2-Colouring	O(1)
2-SAT	$\Omega\left(\frac{\log n}{\log\log n}\right)$, $O(\sqrt{n})$ [Fischer et al.]
3-Colouring, 3-SAT, 3-LIN(p)	$\Omega(n)$ [Ben-Sasson et al.]
Horn 3-SAT	$\Omega(n)$ [Bhattacharyya, Yoshida]

Bhattacharyya and Yoshida have solved this problem over 2 element templates and establish a constant/sublinear/linear trichotomy.

The algebra associated with CSP($\mathbf{A}$)

Remark

The starting point of our investigation of the query complexity of CSP($\mathbf{A}$) is an observation of Yoshida:

> *The query complexity of CSP($\mathbf{A}$) is determined by the algebra of polymorphisms of* $\mathbf{A}$.

Definition

- An operation $f\colon A^k \to A$ is a polymorphism of $\mathbf{A}$ if for each relation (r-ary) R of $\mathbf{A}$ and for all $\vec{s}_1, \ldots, \vec{s}_n \in R$:

$$(f(s_1^1, \ldots, s_n^1), \ldots, f(s_1^r, \ldots, s_n^r)) \in R.$$

- For $\mathbf{A}$ a relational structure, Pol($\mathbf{A}$) denotes the set of polymorphisms of $\mathbf{A}$ and $\mathrm{Alg}(\mathbf{A}) = (A, \mathrm{Pol}(\mathbf{A}))$, the algebra of polymorphisms of $\mathbf{A}$.

Some special polymorphisms

Examples

Let A be a finite set.

- A Maltsev operation on A is a function $p(x,y,z)$ that satisfies the equations
$$p(y,x,x) = p(x,x,y) = y.$$
- A majority operation on A is a function $m(x,y,z)$ that satisfies the equations
$$m(y,x,x) = m(x,y,x) = m(x,x,y) = x.$$
- A near-unanimity operation on A is a function $t(\bar{x})$ that satisfies the equations
$$t(y,x,x,\dots,x,x) = t(x,y,x,\dots,x,x) = \cdots = t(x,x,\dots,x,y) = x$$

Some examples

Examples

- The template $\langle\{0,1\},\neq_2\rangle$ has both Maltsev and majority polymorphisms. CSPs over this template are essentially instances of graph 2-colouring.
- 3-LIN(*p*), the structure over the *p*-element field (*p* a prime) whose relations are all affine subspaces of dimension 2 or 3, has a Maltsev polymorphism, but no majority polymorphism.
- The boolean structure for 2-SAT has a majority polymorphism, but no Maltsev polymorphism.
- The boolean structure for 3-SAT only has trivial polymorphisms.

Constant query testable templates

Theorem (FOCS 2016)

Let $\mathbf{A}$ be a finite structure. CSP$(\mathbf{A})$ is constant query testable if and only if Alg$(\mathbf{A})$ has a Maltsev operation and a majority operation.

Remarks

- *An algebra that has both Maltsev and majority operations is said to generate an arithmetic variety.*
- *A finite product of finite fields is an example of such an algebra.*
- *(Pixley) Having both types of operations is equivalent to having an operation $t(x,y,z)$ that satisfies the equations:*
  $t(y,x,x) = t(x,x,y) = t(y,x,y) = y.$

The non constant query testable case

Remarks

- *An algebra $\mathbb{A}$ that has a Maltsev operation will have a majority operation if and only if it does not (primitive-positive) interpret any module.*
- *If modules can be pp-interpreted by $\mathbb{A}$, then it can be shown that testing CSP$(\mathbf{A})$ requires a linear number of queries (since this is true for linear structures).*
- *If $\mathbb{A}$ fails to have a Maltsev operation, then it interprets a finite structure that has a single binary reflexive, but not symmetric relation.*
- *A modification of an argument from Fischer et al. for 2-SAT can be used to establish an $\Omega\left(\frac{\log n}{\log \log n}\right)$ lower bound on the query complexity of CSP$(\mathbf{A})$.*

The constant query case

Sketch of proof: Pre-processing

- Suppose that **A** has Maltsev and majority polymorphisms and we are given a testing instance: ε, $I = (V, \mathcal{C})$, $w\colon V \to [0,1]$ and query access to an assignment $f\colon V \to A$.
- Using the majority polymorphism, we can produce an equivalent instance $I' = (\mathbb{A}_v, \{((v,w), R_{vw})\})_{v,w \in V}$ whose constraints are all binary and which is $(2,3)$-consistent.
- Using the Maltsev polymorphism, it follows that for variables v, w, the constraint relation $R_{vw} \subseteq \mathbb{A}_v \times \mathbb{A}_w$ is a "thick mapping", i.e., modulo compatible equivalence relations on $\mathbb{A}_v$ and $\mathbb{A}_w$, R_{vw} is the graph of a homomorphism.

The constant query case

Sketch of proof: Reductions

- (Factoring) If for some $v \in V$, the constraints of I' don't distinguish between two elements $a, b \in A_v$, we can produce an "equivalent", reduced instance by factoring out such pairs of elements. The domain $\mathbb{A}_v$ is replaced with a proper quotient of it.
- (Splitting) If some domain $\mathbb{A}_v$ can be represented as a subdirect product of domains $\mathbb{A}_v^1 \times \mathbb{A}_v^2$, then we can replace the variable v with a pair of new variables v^1, v^2 and replace any constraint relation R_{vw} by new relations R_{v^1w} and R_{v^2w} to produce an "equivalent" instance.

Remark

Applying either of these types of reductions will produce an instance whose domains are smaller in size.

The constant query case

Isomorphism reduction

- After repeatedly applying the factoring and splitting reductions our domains will be subdirectly irreducible.
- It follows that for each variable v, there will be some w such that R_{wv} is the graph of a surjective homomorphism.
- v and w will be called equivalent if R_{vw} is the graph of an isomorphism.
- Our final type of reduction is to identity pairs of equivalent variables of I' to produce an "equivalent" instance with fewer variables.
- After applying all three of these types of reductions until they can no longer be applied, we will end up with a trivial instance.

Remark

We show that the query complexity of CSP$(\mathbf{A})$ in this case will be $2^{O(A)}/\varepsilon^2$. The ε-tester produced is 1-sided.

Related result and question

For a structure $\mathbf{A}$, define $\exists\mathrm{CSP}(\mathbf{A})$ to be the set of existentially quantified instances of $\mathrm{CSP}(\mathbf{A})$.

Theorem (FOCS 2016)

For a finite structure $\mathbf{A}$, the query complexity of $\exists CSP(\mathbf{A})$ is:

1. *constant if $\mathbf{A}$ has Maltsev and majority polymorphisms, else is*
2. *sublinear if $\mathbf{A}$ has a near unanimity polymorphism, else is*
3. *linear.*

Question

Does the above trichotomy holds for regular old $\mathrm{CSP}(\mathbf{A})$?