Equivariant algorithms for constraint satisfaction problems over coset templates

Sławomir Lasota

University of Warsaw

Abstract

We investigate the Constraint Satisfaction Problem (CSP) over templates with a group structure, and algorithms solving CSP that are equivariant, i.e. invariant under a natural group action induced by a template. Our main result is a method of proving the implication: if CSP over a coset template $T$ is solvable by an equivariant algorithm then $T$ is 2-Helly (or equivalently, has a majority polymorphism). Therefore bounded width, and definability in fixed-point logics, coincide with 2-Helly. Even if these facts may be derived from already known results, our new proof method has two advantages. First, the proof is short, self-contained, and completely avoids referring to the omitting-types theorems. Second, it brings to light some new connections between CSP theory and descriptive complexity theory, via a construction similar to CFI graphs.

Keywords:

1. Introduction

Many natural computational problems may be seen as instantiations of a generic framework called constraint satisfaction problem (CSP). In a nutshell, a CSP is parametrized by a template, a finite relational structure $T$; the CSP over $T$ asks if a given relational structure $I$ over the same vocabulary as $T$ admits a homomorphism to $T$ (called solution of $I$). For every template $T$, the CSP over $T$ (denoted CSP$(T)$) is always in NP; a famous conjecture due to Feder and Vardi [1] says that for every template $T$, the CSP$(T)$ is either solvable in P, or NP-complete.

We concentrate on coset templates where, roughly speaking, both the carrier set and the relations have a group structure [1]. These templates are cores and admit Malcev polymorphism, and are thus in P [2,3]. A coset template $T$ naturally induces a group action on (partial) solutions. If, roughly speaking, the induced group action can be extended to the state space of an algorithm solving CTP$(T)$, and the algorithm is invariant under the group action, we call the algorithm equivariant.

A widely studied equivariant algorithm is the $(k,l)$-consistency algorithm, where $1 \leq k \leq l$. The algorithm computes a family of partial solutions of size at most $k$, conforming to a local consistency condition applying to subsets of an instance of size at most $l$. A template $T$ has bounded width if CSP$(T)$ is solvable by the $(k,l)$-consistency algorithm, for some $k \leq l$. Another source of examples of equivariant algorithms are logics (via their decision procedures); relevant logics for us will be fix-point extensions of first order logic, like LFP or IFP [4]. We say that CSP$(T)$ is definable in a logic if some formula defines the set of all solvable instances of CSP$(T)$. It is well known that bounded width is equivalent to definability in Datalog [1].

Our technical contribution is the proof of the following implication: if a coset template $T$ has bounded width then $T$ is 2-Helly. The 2-Helly property says that for any partial solution $h$ of an instance $I$, if $h$ does not extend to a solution of $I$ then the restriction of $h$ to some two elements of its domain does not either. This is a robust property of templates with many equivalent characterizations (e.g. strict width 2, or existence of a majority polymorphism) [1]. Our proof method is however not specific for bounded width, and can be straightforwardly adapted to any other algorithm which is local and equivariant, and therefore if CSP$(T)$ is solvable by a local equivariant algorithm then $T$ is 2-Helly. This implies equivalence of the following conditions for coset templates:

$$2\text{-Helly} \iff \text{bounded width} \iff \text{definability in LFP}.$$  

(1)

These equivalences are not a new result: the first one may be inferred e.g. from Lemma 9 in [5] (even for all core templates with Malcev polymorphism), while the second one follows from [6] together with the results of [7] (cf. also [8]). All these results build on Tame Congruence Theory [8], and their proofs are a detour through the deep omitting-type theorems, cf. [10]. Contrarily to this, our proof has an advantage of being short, elementary, and self-contained, thus offering a direct insight into the problem. Furthermore, our result is more general than (1): it implies that all local and equivariant algorithms that can capture 2-Helly templates are equally expressive, namely they solve CSP$(T)$ for exactly the same coset templates $T$.

Finally, our proof brings to light interesting connections between the CSP theory and the descriptive complexity theory: the crucial step of the proof is essentially based on a construction similar to CFI graphs, the intricate construction of Cai, Fürer and Immerman [11]. CFI graphs have been designed to separate properties of relational structures decidable in polynomial time from the logic IFP+C (IFP with counting quantifiers).

Preprint submitted to Elsevier

March 3, 2015

Supported by the NCN grant 2012/07/B/ST6/01497.
A similar construction has been used later in [12] to show lack of determination of Turing machines in set with atoms [13]. The crucial step of our proof is actually a significant generalization of the construction of [12].

For completeness we mention a recent paper of Barto [14] which announces the collapse of bounded width hierarchy for all templates: bounded width implies width (2, 3), which is however weaker than 2-Helly in general.

2. Preliminaries

2.1. Constraint satisfaction problems

A template $T$ is a finite relational structure, i.e. consists of a finite carrier set $T$ (denoted by the same symbol as a template) and a finite family of relations in $T$. Each relation $R \subseteq T^n$ is of a specified arity, $arity(R) = n$. Let $T$ be fixed henceforth.

An instance $I$ over a template $T$ consists of a finite set $I$ of elements, and a finite set of constraints. A constraint, written $R(a_1, \ldots, a_n)$, is specified by a template relation $R$ and an $n$-tuple of elements of $I$, where $arity(R) = n$.

A partial function $h$ from $I$ to $T$, with $[a_1, \ldots, a_n] \subseteq \text{dom}(h)$, satisfies a a constraint $R(a_1, \ldots, a_n)$ in $I$ when $R(h(a_1), \ldots, h(a_n))$ holds in $T$. If $h$ satisfies all constraints in its domain, $h$ is a partial solution of $I$, and $h$ is a solution when it is total. By the size of a partial solution $h$ we mean the size of $\text{dom}(h)$. The constraint satisfaction problem over $T$, denoted CSP($T$), is a decision problem that asks if a given instance over $T$ has a solution.

There are many equivalent formulations of the problem. For instance, one can see $I$ and $T$ as relational structures over the same vocabulary, and then CSP($T$) asks if there is a homomorphism from $I$ to $T$.

2.2. Bounded width

Consider an instance $I$ and a family $\mathcal{H}$ of its partial solutions of size at most $k$, for some $k > 0$. It will be convenient to split $\mathcal{H}$ into the subfamilies $\mathcal{H}_X$, where $\mathcal{H}_X = \{h \in \mathcal{H} : \text{dom}(h) = X\}$. Fix $k$ and $l \geq k$, and consider two subsets $X \subseteq Y$ of $I$ of size $k$ and $l$, respectively. A partial solution $h \in \mathcal{H}_X$ is consistent wrt. $\mathcal{H}$ and $(X, Y)$ if

$h$ extends to a partial solution $h'$ with $\text{dom}(h') = Y$, whose restriction $h'|_X$ to every subset $X' \subseteq Y$ of size at most $k$ belongs to $\mathcal{H}_X$.

The $(k, l)$-consistency algorithm takes as input an instance $I$ over $T$, and computes the greatest family $\mathcal{H}$ of partial solutions of size at most $k$, such that every $h \in \mathcal{H}$ is consistent wrt. $\mathcal{H}$ and $(X, Y)$, for every $X$ and $Y$ as above.

The algorithm starts with $\mathcal{H}$ containing all partial solutions of size $k$, and proceeds by iteratively removing from $\mathcal{H}$ partial solutions $h$ that falsify the consistency condition. The order of removing is irrelevant, but for convenience we assume some fixed enumeration $(X_1, Y_1), \ldots, (X_n, Y_n)$ of all pairs $(X, Y)$ of subsets of $I$ as above. Furthermore we assume that the algorithm proceeds by iteratively executing the following subroutine until stabilization:

for $i = 1, 2, \ldots, n$, 
\[ \mathcal{H}_X := \{h \in \mathcal{H}_X : h \text{ is consistent wrt. } \mathcal{H} \text{ and } (X_i, Y_i)\} \]  

Every single update ($*$) of $\mathcal{H}_X$, for a fixed pair $(X, Y)$, we call a stage.

When the stabilization is reached and $\mathcal{H}$ is nonempty then all the subfamilies $\mathcal{H}_X$ are also nonempty. The $(k, l)$-consistency algorithm answers positively if the family $\mathcal{H}$ computed by the algorithm is nonempty; otherwise, the algorithm answers negatively.

If an input instance $I$ is solvable (i.e. admits a solution) then the $(k, l)$-consistency algorithm answers positively, for every $k \leq l$. We say that a template $T$ has width $(k, l)$ if the $(k, l)$-consistency algorithm correctly solves CSP($T$), namely the algorithm answers positively only if the input instance $I$ is solvable. $T$ has bounded width if it has some width $(k, l)$.

2.3. 2-Helly templates

**Definition 2.1.** For an instance $I$ over some template, and $k < j$, a $(k, j)$-anomaly is a partial solution $h$ of $I$ of size $j$ that does not extend to a solution, such that restriction of $h$ to every $k$-element subset of $\text{dom}(h)$ does extend to a solution.

Clearly a $(k, j)$-anomaly is also $(k', j)$-anomaly, for $k' < k$.

**Definition 2.2.** A template $T$ is 2-Helly if no instance of $T$ admits a $(2, j)$-anomaly, for $j > 2$.

In other words: for every partial solution $h$ of size $j > 2$, if the restriction of $h$ to every 2-element subset of its domain extends to a solution then $h$ does extend to a solution too. Analogously one may define $k$-Helly for arbitrary $k$, which however will not be needed here.

We conveniently characterize 2-Helly templates as follows.

**Lemma 2.3.** A template $T$ is 2-Helly iff no instance of $T$ admits a $(k, k + 1)$-anomaly, for $k \geq 2$.

**Proof.** For one direction, we observe that a $(k, k + 1)$-anomaly is also a $(2, k + 1)$-anomaly.

For the other direction, consider an instance with some fixed $(2, j)$-anomaly $h$, for $j > 2$. For every subset $X \subseteq \text{dom}(h)$, the restriction $h|_X$ either extends to a solution of $I$, or not. Consider the minimal subset $X$ wrt. inclusion such that $h|_X$ does not extend to a solution of $I$. For all strict subsets $X' \subseteq X$, $f|_{X'}$ extends to a solution, hence $f|_X$ is a $(k - 1, k)$-anomaly, where $k$ is the size of $X$. Note that $k > 2$.

2.4. The pp-definable relations

We adopt the convention to mention explicitly the free variables of a formula $\phi(x_1, \ldots, x_n)$. In the specific instances $I$ of CSP($T$) used in our proof it will be convenient to use pp-definable relations, i.e. relations definable by an existential first-order formula of the form:

$$\phi(x_1, \ldots, x_n) \equiv \exists x_{n+1}, \ldots, x_{n+m}. \psi_1 \land \ldots \land \psi_l$$  

(2)
where every subformula $\psi_i$ is an atomic proposition $R(x_1, \ldots, x_n)$, for some template relation $R$. The formula $\phi$ defines the $n$-ary relation in $T$ containing the tuples

$$(t_1, \ldots, t_n) \in T^n$$

such that the valuation $x_i \mapsto t_i$, $x_n \mapsto t_n$ satisfies $\phi$. The pp-definable relations are closed under projection and intersection.

In the sequel we feel free to implicitly assume that elements of an instance are totally ordered. The implicit order allows us to treat (partial) solutions as tuples, and allows to state the following useful fact:

**Fact 2.4.** Let $X \subseteq I$ be a subset of an instance. The set of partial solutions with domain $X$ that extend to a solution of $I$, if nonempty, is pp-definable.

We recall the following widely known fact:

**Fact 2.5.** Adding a pp-definable relation to a template yields a computationally equivalent template. In particular, bounded width and 2-Helly are preserved.

2.5. Almost-direct product of groups

Overloading the notation, we write 1 for the identity element in any group. We use the diagrammatic order for writing the group operation $\tau \pi$ on elements $\tau, \pi$ of a group.

In the proof we will need the following elementary notion from group theory.

**Definition 2.6.** Let $G_1$, $G_2$ and $G_3$ be arbitrary finite groups and let $H \leq G_1 \times G_2 \times G_3$ be a subgroup of the direct product. We call $H$ an almost-direct product of $G_1, G_2, G_3$ if $H$ verifies the following conditions:

1. $H \neq G_1 \times G_2 \times G_3$ \hspace{1cm} (3)
2. $\forall \pi_2 \in G_2, \pi_3 \in G_3, \exists \pi_1 \in G_1, (\pi_1, \pi_2, \pi_3) \in H$ \hspace{1cm} (4)
3. $\forall \pi_1 \in G_1, \pi_2 \in G_2, \exists \pi_3 \in G_3, (\pi_1, \pi_2, \pi_3) \in H$ \hspace{1cm} (5)
4. $\forall \pi_1 \in G_1, (\pi_2, \pi_3) \in G_2 \times G_3, \exists \pi_1 \in G_1, (\pi_1, \pi_2, \pi_3) \in H$ \hspace{1cm} (6)

Furthermore, an almost-direct product $H$ is strict if $\pi_1$ (resp. $\pi_2$, $\pi_3$) in condition (4) (resp. (5), (6)) is uniquely determined.

**Lemma 2.7.** Every almost-direct product $H$ has a surjective homomorphism onto a strict almost-direct product.

**Proof.** Let $H \leq G_1 \times G_2 \times G_3$ be an almost-direct product. Consider the following subgroup $N_1$ of $G_1$:

$$N_1 = \{ \pi_1 \in G_1 : (\pi_1, 1, 1) \in H \}$$

and observe that $N_1$ is a normal subgroup of $G_1$. Likewise define the normal subgroups $N_2$ and $N_3$ of $G_2$ and $G_3$, respectively. In consequence, the product $N = N_1 \times N_2 \times N_3$ is not just a subgroup of $H$ (which follows by the very definition of $N_1, N_2, N_3$), but actually a normal subgroup of $H$. Define the groups $[G_1], [G_2], [G_3]$ and $[H]$ as the quotients by $N_1, N_2, N_3$ and $N$, respectively.

By the definition of $N_1$, the quotient group $[G_1]$ is obtained by identifying its elements $\pi_1, \pi_1'$ that are equivalent:

$$\pi_1 \equiv \pi_1' \Leftrightarrow (\forall \pi_2, \pi_3, (\pi_1, \pi_2, \pi_3) \in H \Leftrightarrow (\pi_1', \pi_2, \pi_3) \in H).$$

Similarly one defines the equivalences $\equiv_2$ and $\equiv_3$. Note that $H$ is closed under the three equivalences; for instance, $(\pi_1, \pi_2, \pi_3) \in H$ and $\pi_1 \equiv \pi_1'$ implies $(\pi_1', \pi_2, \pi_3) \in H$.

$[H]$, being the quotient of $H$, is an almost-direct product of $[G_1], [G_2], [G_3]$. We claim that $[H]$ is strict. Concentrating on point (4) in Definition 2.6 (the remaining two conditions are treated similarly), we need to prove uniqueness of $\pi_1$. Suppose

$$(\pi_1, \pi_2, \pi_3) \in [H] \text{ and } (\pi_1', \pi_2, \pi_3) \in [H];$$

we need to derive $\pi_1 = \pi_1'$. As $H$ is closed under the three equivalences, there are some $\rho_1, \rho_1', \rho_2, \rho_3$ such that

$$\tau = (\rho_1, \rho_2, \rho_3) \in H, \quad \tau' = (\rho_1', \rho_2, \rho_3) \in H,$$

and (writing $[\rho]$ for the equivalence class containing $\rho$)

$$[\rho_1] = \pi_1, \quad [\rho_1'] = \pi_1', \quad [\rho_2] = \pi_2, \quad [\rho_3] = \pi_3;$$

and we need to derive $\rho_1 \equiv \rho_1'$. The equivalence follows easily: whenever $\sigma = (\pi_1, \tau_2, \tau_3) \in H$, we have

$$(\pi_1', \tau_2, \tau_3) = \sigma^{-1} \tau' \in H.$$ 



**Lemma 2.8.** Every strict almost-direct product is commutative.

**Proof.** Let $H \leq G_1 \times G_2 \times G_3$ be a strict almost-direct product and let $\pi, \tau \in G_1$. We know that there exist $\rho_2 \in G_2$ and $\rho_3 \in G_3$ so that (we do not use the uniqueness of $\rho_2$ and $\rho_3$ here):

$$([\pi, \rho_2], \rho_3) \in H \quad \text{and} \quad (\pi, [\rho_2, \rho_3]) \in H.$$

Applying the group operation to these two elements in two different orders we get:

$$([\pi, \rho_2], \rho_3) \in H \quad \text{and} \quad (\pi, [\rho_2, \rho_3]) \in H.$$

Now using the uniqueness of $\pi \tau$ and $\tau \pi$, we deduce that $\pi \tau = \tau \pi$. As $\pi$ and $\tau$ have been chosen arbitrarily, the group $G_1$ is commutative. Likewise for $G_2$ and $G_3$, and in consequence also for the subgroup $H \leq G_1 \times G_2 \times G_3$.

3. Coset templates

Below by a coset we always mean a right coset. (This choice is however arbitrary and we could consider left cosets instead.)

**Definition 3.1.** Coset templates are particular templates $T$ that satisfy the following conditions:

1. The notion seems to be of independent interest; it is related to the arity of a permutation group, as investigated for instance by Cherlin at al. in [1].
2. I am grateful to Szymek Toruńczyk for the simplification of the proof.
- the carrier set of $T$ is a disjoint union of groups, call these groups carrier groups;
- every $n$-ary relation $R$ in $T$ is a coset in the direct product $G_1 \times \ldots \times G_n$ of some carrier groups $G_1, \ldots, G_n$;
- for a relation $R \subseteq G_1 \times \ldots \times G_n$ in $T$ and $\pi \in G_1 \times \ldots \times G_n$, the coset $R \pi$ is also a relation in $T$;
- for every carrier group $G$, $T$ has a unary relation $\{1\}$ containing exactly one element, the identity of $G$.

Note that the last two conditions imply that a coset template contains every singleton as a unary relation, and thus is a rigid core, i.e., admits no nontrivial endomorphisms.

**Example 3.1.** Here is a family of coset templates $T_n$, for $n \geq 2$. The carrier set of $T_n$ is $[1, n]$, the cyclic group of order 2. Relations of $T_n$ are, except the two singleton unary relations $1(\_)$ and $\pi(\_)$, two $n$-ary relations

$$R_{\text{even}}, R_{\text{odd}} \subseteq T^n$$

containing $n$-tuples where $\pi$ appears an even (resp. odd) number of times. Template $T_2$ is 2-Helly, while for $n > 2$, template $T_n$ is not. Indeed, a $(2,3)$-anomaly is admitted by an instance over $T_3$, consisting of three elements $a_1, a_2, a_3$ and four constraints:

$$\pi(a_1) \quad \pi(a_2) \quad \pi(a_3) \quad R_{\text{even}}(a_1, a_2, a_3).$$

Consider a relation $R \subseteq G_1 \times \ldots \times G_n$ in a coset template, and an instance $I$. For a constraint $R(a_1, \ldots, a_n)$ in $I$ and $i \in \{1, \ldots, n\}$, we call $G_i$ a constraining group of $a_i$. In order to have a solution, an instance $I$ has to be non-contradictory, in the sense that every element must have exactly one constraining group (elements with no constraining group may be safely removed from $I$). We only consider non-contradictory instances from now on.

Consider a fixed coset template $T$ and an instance $I$ over $T$. By a pre-solution of $I$ we mean any function $s : I \to T$ that maps every element $i \in I$ to an element of the constraining group of $i$. Pre-solutions of an instance $I$ form a group, with group operation defined point-wise. One can also speak of partial pre-solutions, whose domain is a subset of $I$. Using an implicit order of elements of an instance, (partial) pre-solutions of $I$ are elements of the direct product of constraining groups of some elements of $I$.

We distinguish subgroup instances, where all relations $R$ appearing in the constraints $R(a_1, \ldots, a_n)$ are subgroups, instead of arbitrary cosets.

**Fact 3.2.** (1) The set of all solutions $\mathcal{H}$ of an instance $I$, if nonempty, is a coset in the group of pre-solutions. (2) In consequence, if $I$ is a subgroup instance then $\mathcal{H}$ is a subgroup of the group of pre-solutions.

**Proof.** To show (1) observe that for every constraint $c = R(a_1, \ldots, a_n)$ in $I$, the set of all pre-solutions $\mathcal{H}_c$ satisfying that particular constraint is a coset in the group of pre-solutions. As solutions $\mathcal{H}$ are exactly the intersection,

$$\mathcal{H} = \bigcap_c \mathcal{H}_c,$$

for $c$ ranging over all constraints in $I$, by closure of cosets under nonempty intersection we derive that $\mathcal{H}$ is a coset.

(2) follows by an observation that the tuple $(1, \ldots, 1)$ of identities is always a solution, in case of a subgroup instance. □

Wlog. we may assume that every variable in a formula specifying a pp-definable set, appears in some atomic proposition (intuitively, the variable is constrained). Then every pp-definable relation is essentially a projection of the set of solutions of some instance (variables are element of the instance, and atomic propositions are its constraints), and by Fact 3.2 we derive the following corollary:

**Fact 3.3.** (1) Every pp-definable relation $R \subseteq G_1 \times \ldots \times G_n$ in $T$ is a coset in $G_1 \times \ldots \times G_n$. (2) If $R$ is pp-definable and $\pi \in G_1 \times \ldots \times G_n$ then $R \pi$ is pp-definable as well.

### 3.1. Action of pre-solutions

For a fixed instance $I$, define the action of pre-solutions on (partial) pre-solutions (thus in particular on (partial) solutions). For a (partial) pre-solution $h : I \to T$ and a pre-solution $s$, let $h \cdot s$ be defined by the point-wise group operation:

$$(h \cdot s)(a) = h(a) s(a), \quad \text{for } a \in \text{dom}(h).$$

We will apply the action to the instance $I$ itself; let $I \cdot s$ be an instance with the same carrier set as $I$, whose constraints are obtained from the constraints of $I$ as follows: for every constraint $R(a_1, \ldots, a_n)$ of $I$, the instance $I \cdot s$ contains a constraint

$$R'(a_1, \ldots, a_n), \quad \text{where } R' = R(s(a_1), \ldots, s(a_n)).$$

As $R$ is a coset in the direct product of constraining groups of $a_1, \ldots, a_n$, so is $R'$. Note that the action preserves constraining groups, and hence pre-solutions, of an instance.

It is important to notice that solutions are invariant under the action of pre-solutions:

**Fact 3.4.** If $h$ is a solution of $I$ then $h \cdot s$ is a solution of $I \cdot s$.

Recall that the $(k, l)$-consistency algorithm proceeds in stages, where every stage applies to a pair of subsets $(X, Y)$ of an instance; and that the order of stages is induced by a fixed enumeration of all pairs $(X, Y)$. Assume wlog. that the enumeration only depends on the size of the input $I$, but not on the constraints in $I$. This guarantees that the $(k, l)$-consistency algorithm is invariant under the action of pre-solutions (we write $\mathcal{H}_k(I, i)$ for the value of $\mathcal{H}_k$ after the $i$th stage, for an input instance $I$):

**Fact 3.5.** The $(k, l)$-consistency algorithm is invariant under the action of pre-solutions: for every instance $I$ and its pre-solution $s$, after every $i$th stage of the algorithm,

$$\forall X, \quad \mathcal{H}_k(I, i, s) = \mathcal{H}_k(I, i) \cdot s$$

(the latter action of $s$ is defined as the direct image).
4. The result

As our main technical result we prove:

**Theorem 4.1.** For coset templates, bounded width implies (and is thus equivalent to) 2-Helly.

The proof is self-contained and short (cf. Section 5). In the proof we construct, assuming a template \( T \) is not 2-Helly, a family of instances that are hard for the consistency algorithm. Interestingly, the hard instances are a generalization of CFI procedures of) natural logics, including first-order logic, extensions thereof with fixpoints, second-order logic, etc.

Claim 4.1.1. Let \( T \) be a coset template. If CSP(\( T \)) is solvable by an algorithm that is local and equivariant, then \( T \) is 2-Helly.

In particular, coset templates \( T \) for which CSP(\( T \)) is definable in LFP (first-order logic with fixpoints), but also in more expressive logics like IFP, IFP+C, or even second-order logic, are exactly 2-Helly templates. Therefore LFP is equivalent to bounded width for coset templates. Again, this is not a new result: using the omitting-types theorems one can derive this equivalence from \([8]\) and \([9]\) for all templates.

5. Proof of Theorem 4.1

Fix a coset template \( T \) and suppose it is not 2-Helly. We aim at showing that \( T \) has no bounded width. We start with the following claim, whose proof is postponed to Section 5.1.

**Proposition 5.1.** There are some subgroups \( S_1, S_2, S_3 \) of carrier groups, and an almost-direct product

\[
R \leq S_1 \times S_2 \times S_3
\]

such that \( R \) and all its cosets in \( S_1 \times S_2 \times S_3 \) are pp-definable (as ternary relations).

We will now define a class of instances, called \( n \)-torus instances, and then show that the consistency algorithm yields incorrect results for these instances. An \( n \)-torus instance is an instance of particular shape. It contains exactly \( 3n^2 \) elements, say

\[
a_{ij}, b_{ij}, c_{ij},
\]

for \( i, j \in \{0 \ldots n-1\} \), and exactly \( 2n^2 \) constraints:

\[
R_i(a_{ij}, b_{ij}, c_{ij}) \quad \text{and} \quad R_i'(a_{i(i+1)}, b_{i(i+1)}, c_{ij}),
\]

for \( i, j \in \{0 \ldots n-1\} \). Relations \( R_i \) and \( R_i' \) are arbitrary cosets of \( R \) in \( S_1 \times S_2 \times S_3 \) (by Proposition 5.1 they are all pp-definable). We rely here on Fact 2.5. We adopt the convention that all indices are counted modulo \( n \), e.g. \( a_{in} = a_0 \) and \( a_{ni} = a_0 \). The \( 2n^2 \) tuples

\[
(a_{ij}, b_{ij}, c_{ij}) \quad \text{and} \quad (a_{i(i+1)}, b_{i(i+1)}, c_{ij})
\]

appearing in constraints (8) we call positions of an \( n \)-torus. The shape of a 3-torus instance is depicted below, with edges representing elements, and triangles representing positions.

```
Every element a of an n-torus appears in exactly two constraints. Thus every position (a, b, c) has exactly three neighbors, namely those other positions that contain any of a, b, c.
```
This defines the neighborhood graph, whose vertices are positions of an n-torus. The graph is 3-regular.

The n-torus instances are built from triangulations of a torus surface. It is however not particularly important to use a torus; equally well a sphere could be used instead, or any other connected closed surface, as long as, intuitively speaking, the surface is hard to decompose into small pieces. The property can be formally stated as follows:

**Fact 5.2.** After removing $j < n$ positions, the neighborhood graph of an n-torus still contains a connected component of size at least $2n^2 - j^2$.

Consider the $(k, l)$-consistency algorithm, for fixed $k \leq l$.

**Definition 5.3.** Let $I$ be an n-torus, $(a, b, c)$ a position of $I$, $X$ a set of at most $k$ elements of $I$, and $i \geq 0$. We say that the $(k, l)$-consistency algorithm ignores $(a, b, c)$ at $X$ after the $ith$ stage if 

$$\mathcal{H}_X(I, i) = \mathcal{H}_X(I', i),$$

for every n-torus $I'$ that differs from $I$ only by one constraint at position $(a, b, c)$.

**Proposition 5.4.** There is some $m \in \mathbb{N}$, depending only on $l$ such that for every n-torus $I$ with a sufficiently large $n$, the $(k, l)$-consistency algorithm ignores, at every $X$ and after every stage, all but at most $m$ positions of $I$.

Therefore for every instance, the $(k, l)$-consistency algorithm necessarily ignores some position after the last stage, which easily entails incorrectness of the algorithm. Indeed, consider an n-torus $I_k$ that uses the relation $R$ of Proposition 5.1 in all its constraints. $I_k$ is solvable and hence the $(k, l)$-consistency algorithm answers positively. On one hand, by Proposition 5.3 there is some position $(a_0, b_0, c_0)$ such that the $(k, l)$-consistency algorithm answers positively for the instance obtained by replacing the relation $R$ in the constraint $R(a_0, b_0, c_0)$ in $I_k$ with any other coset of $R$ in $S_1 \times S_2 \times S_3$. On the other hand, we prove:

**Proposition 5.5.** Replacing the relation $R$ with any other coset of $R$ in one constraint in $I_k$ yields an unsolvable instance.

In consequence, the $(k, l)$-consistency algorithm is incorrect. This completes the proof of Theorem 5.1 once we prove the three yet unproved claims, namely Propositions 5.1, 5.4, and 5.5.

### 5.1. Proof of Proposition 5.7

By Lemma 5.3 some instance $I$ contains a $(k, k + 1)$-anomaly, for $k \geq 2$. Note that this implies that this instance has at least one solution.

Wlog. we can assume that $I$ is a subgroup instance. Indeed, for an arbitrary solution $h$ of $I$, define a new instance by the action of $h^{-1}$:

$$I' := I \cdot h^{-1}.$$  

As $h$ is a solution of $I$, for every constraint $R(a_1, \ldots, a_n)$ in $I$, the tuple $(h(a_1), \ldots, h(a_n))$ is in $R$. Hence every relation appearing in a constraint of $I'$ is a subgroup in the product of constraining groups, as required. Due to Fact 5.4 an anomaly admitted by $I$ translates to an anomaly admitted by $I'$.

Proposition 5.1 follows from the following two lemmas.

**Lemma 5.6.** If some subgroup instance admits a $(k, k + 1)$-anomaly, for $k \geq 2$, then some subgroup instance admits a $(k - 1, k)$-anomaly.

Hence some instance admits a $(2, 3)$-anomaly.

**Lemma 5.7.** If some subgroup instance admits a $(2, 3)$-anomaly then there are subgroups $S_1, S_2, S_3$ of carrier groups, and a pp-definable almost-direct product

$$R \leq S_1 \times S_2 \times S_3.$$  

Using Fact 5.3 we deduce that all cosets of $R$ are pp-definable too, as required by Proposition 5.1. It only remains to prove Lemmas 5.6 and 5.7.

**Proof of Lemma 5.6** Fix a $(k, k + 1)$-anomaly $h$ in a subgroup instance $I$, for some $k \geq 2$, and choose an arbitrary element $a \in \text{dom}(h)$. Let $X = \text{dom}(h) \setminus \{a\}$. Define the new instance $I'$, with the same domain as $I$, whose constraints are all constraints of $I$ plus one additional unary constraint $1(a)$ requiring that $a$ should be mapped to the identity in its constraining group.

As $h$ is an anomaly, the restriction $h_{|X}$ extends to a solution of $I$, i.e. $I$ has a solution $\tilde{h}$ satisfying $\tilde{h}(a) = h(a)$. Using an arbitrary such solution we define another partial solution $h'$ of $I$:

$$h'(x) = h(x) \cdot \tilde{h}^{-1}(x)$$  

with $\text{dom}(h') = \text{dom}(h) = X \cup \{a\}$.

Consider the restriction $h'' = h'|_X$. We claim that $h''$ is a $(k - 1, k)$-anomaly in $I'$. Indeed, for every subset $X' \subseteq X$ of size $k - 1$, $h''|_{X'\setminus\{a\}}$ extends to a solution of $I$, hence $h''|_{X'\setminus\{a\}}$ also extends to a solution of $I'$, and hence $h''$ also extends to a solution of $I'$, Moreover, $h$ does not extend to a solution of $I$, hence $h''$ does not extend to a solution of $I'$, and thus $h''$ also does not extend to a solution of $I'$, as every solution of $I'$ is forced to map $a$ to 1.

**Proof of Lemma 5.7** Suppose a subgroup instance $I$ admits a $(2, 3)$-anomaly $h = (\pi_1, \pi_2, \pi_3)$.

Consider the set $H$ of all those partial solutions, with the same domain as $h$, that extend to a solution of $I$. $H$ is a pp-definable ternary relation according to Fact 2.4. By Fact 2.1 we know that $H$ is a subgroup in the product $G_1 \times G_2 \times G_3$ of some three carrier groups. As $h$ is a $(2, 3)$-anomaly, we know (we prefer below to write $H(\pi_1, \pi_2, \pi_3)$ instead of $(\pi_1, \pi_2, \pi_3) \in H)$:

\begin{align*}
\neg H(\pi_1, \pi_2, \pi_3) & \quad (9) \\
\exists r \in G_1. H(r, \pi_2, \pi_3) & \quad (10) \\
\exists r \in G_2. H(\pi_1, r, \pi_3) & \quad (11) \\
\exists r \in G_3. H(\pi_1, \pi_2, r) & \quad (12)
\end{align*}
Now we are ready to define an almost-direct product \( R \leq S_1 \times S_2 \times S_3 \). The subgroups \( S_1 \leq G_1, S_2 \leq G_2 \) and \( S_3 \leq G_3 \) we define as follows:

\[
\begin{align*}
\tau_1 &\in S_1 \quad \iff \exists \rho. H(\tau_1, 1, 1) \wedge \exists \rho. H(\tau_1, 1, \tau) \\
\tau_2 &\in S_2 \quad \iff \exists \rho. H(\tau_2, 1, 1) \wedge \exists \rho. H(1, \tau_2, \tau) \\
\tau_3 &\in S_3 \quad \iff \exists \rho. H(1, \tau, \tau_3) \wedge \exists \rho. H(1, \tau, 1)
\end{align*}
\]

and the subgroup \( R \) as the restriction of \( H \) to \( S_1 \times S_2 \times S_3 \):

\[
R := H \cap S_1 \times S_2 \times S_3.
\]

By the very definition, \( S_1, S_2, S_3 \) and \( R \) are pp-definable.

We need to show the conditions (3)–(6) in Definition 2.6. For (3) (i.e., \( R \neq S_1 \times S_2 \times S_3 \)) we use (9) and (10) to conclude that for \( \tau_1 = \tau^{-1} \pi_1 \in G_1 \) it holds

\[
\neg H(\tau_1, 1, 1).
\]

Moreover, using (10) and (11) we deduce that for some \( \tau \in G_2 \)

\[
H(\tau_1, \tau, 1);
\]

and similarly, using (10) and (12), we deduce that for some \( \tau \in G_3 \)

\[
H(1, \tau_2, \tau).
\]

Thus \( \tau_1 \in S_1 \) and therefore \( (\tau_1, 1, 1) \in S_1 \times S_2 \times S_3 \setminus R \).

Now we concentrate on condition (5) in Definition 2.6 (the remaining two conditions (4) and (5) are shown in the same way). Let \( \tau_1 \in S_1 \) and \( \tau_2 \in S_2 \). By the very definition of \( S_1 \) and \( S_2 \) we know that

\[
H(\tau_1, 1, \tau) \quad \text{and} \quad H(1, \tau_2, \tau');
\]

for some \( \tau, \tau' \in G_3 \). Therefore \( H(\tau_1, \tau_2, \tau \tau') \) and it only remains to show that \( \tau \tau' \in S_3 \). Consider \( \tau (\tau') \) is treated analogously) in order to show \( \tau \in S_3 \). We know already that \( H(\tau_1, 1, \tau) \) which proves a half of the defining condition for \( \tau \in S_3 \). In order to prove the other half, we use the fact that

\[
H(\tau_1, \tau, 1)
\]

for some \( \tau \in G_2 \), to deduce from \( H(\tau_1, 1, \tau) \) that

\[
H(1, \tau^{-1}, \tau),
\]

which completes the proof of \( \tau \in S_3 \).

\[\square\]

5.2. Proof of Proposition 5.4

We will need the following property of almost-direct products:

**Lemma 5.8.** Every coset \( R' \) of \( R \) in \( S_1 \times S_2 \times S_3 \) contains elements of the form

\[
(\tau_1, 1, 1), \quad (1, \tau_2, 1), \quad (1, 1, \tau_3),
\]

for some \( \tau_1 \in S_1, \tau_2 \in S_2 \) and \( \tau_3 \in S_3 \).

Proof. Indeed, let \( \pi = (\pi_1, \pi_2, \pi_3) \in R' \). Knowing that \( \rho = (\rho_1, \pi_2, \pi_3) \in R \) for some \( \rho_1 \in S_1 \), we get

\[
\rho^{-1} \pi = (\rho_1^{-1} \pi_1, 1, 1) \in R'
\]

as required. Likewise one proves the remaining two claims. \[\square\]

The proof of Proposition 5.4 proceeds by induction on the number of stages of the \((k,l)\)-consistency algorithm. The base case concerns the initial value of \( H \); the algorithm ignores at \( X \) all constraints of the input instance not referring to any of elements of \( X \), that is all but at most \( 2k \) constraints.

For the induction step, consider the \( i \)th stage of the algorithm, \( i \geq 1 \). The exact value of \( m \) will be given only later. We only need to consider the unique set \( H_k(I, i) \) that changes in this stage. According to the consistency condition tested by the algorithm, there is a set \( Y \) of \( l \) elements of \( I \) such that the new value of \( H_k \) depends only on \( o(k,l) \) values of \( H_\ell \), for \( X' \subseteq Y \) (cf. (11)). Using the induction assumption for the previous stage and all sets \( X' \), we learn that there are at most \( o(k,l) \cdot m \) constraints not ignored after stage \( i \) at \( X \). We need to show, however, that there are at most \( m \) such constraints. The argument has a geometric flavor, and builds on Fact 5.2. Define \( j = 2k \), the maximal number of constraints referring to elements of \( X \). We now reveal the value of \( m \); we put \( m = j^2 \).

Assume now that \( n \) is sufficiently large; specifically, we need that \( 2n^2 > o(k,l) \cdot m + m \). By Fact 5.2, there is a connected subset \( C \) of positions of size at least \( 2n^2 - m \), so it is larger than \( o(k,l) \cdot m \). By the induction assumption, some position in \( C \) is ignored at \( X \) after \( i \)th stage. For the proof of Proposition 5.4, it is enough to prove that every position in \( C \) is ignored at \( X \) after \( i \)th stage. To this end, since \( C \) is connected, it is now enough to show:

**Claim 5.8.1.** If some position in \( C \) is ignored at \( X \) after the \( i \)th stage, then every neighbor of that position in \( C \) also is.

To show the last claim, consider two neighboring constraints in \( C \), say \( U(a, b, c) \) and \( U'(a, b', c') \), both referring to an element \( a \). Supposing that \( (a, b, c) \) is ignored, we need to demonstrate that \( (a, b', c') \) is ignored too. Let \( \overline{T} \) be an \( n \)-torus obtained from \( I \) by replacing the constraint \( U'(a, b', c') \) with \( \overline{U} = U(a, b', c') \), for some coset

\[
\overline{U} = U'.
\]

We need to show

\[
H_k(I, i) = H_k(\overline{T}, i).
\]

Using Lemma 5.8 we may assume wlog. that \( \pi = (\pi_1, 1, 1) \) for some \( \pi_1 \in S_1 \). Let \( s \) be a pre-solution defined by

\[
s(x) = \begin{cases} 
\pi_1 & \text{if } x = a \\
1 & \text{otherwise.}
\end{cases}
\]

Knowing that \( (a, b, c) \) is ignored, we may write

\[
H_k(I, i) = H_k(\overline{T}, i),
\]

as required. Likewise one proves the remaining two claims. \[\square\]
where the \( n \)-torus \( \mathbb{T} \) is obtained from \( I \) by replacing the constraint \( U(a, b, c) \) with \( T(U(a, b, c)) \), for
\[
T = U^{-1}.
\]

Observe the equality
\[
\mathbb{T} = \mathbb{T} \cdot s.
\]

Using this equality, instead of (13) we will prove
\[
\mathcal{H}_X(I, i) = \mathcal{H}_X(\mathbb{T} \cdot s, i).
\]

The last equality follows easily from the following two equalities:
\[
\mathcal{H}_X(I, i) = \mathcal{H}_X(I \cdot s, i) \quad \mathcal{H}_X(I \cdot s, i) = \mathcal{H}_X(\mathbb{T} \cdot s, i).
\]

The first one follows by Fact 5.5 since \( s \) is identity on all elements of \( X \). Furthermore, the consistency algorithm is invariant under the action of pre-solutions (cf. Fact 5.5), which implies that the second equality above easily follows from (14). This completes the proof of (13).

5.3. Proof of Proposition 5.5

For a position \((a, b, c)\) and a pre-solution \( h \) of \( I \) we write \( h(a, b, c) \) instead of \((h(a), h(b), h(c))\).

Fix a position \((a_0, b_0, c_0)\). Let \( I_R \) be the instance obtained from \( I_R \) by removing the constraint \( R(a_0, b_0, c_0) \). We will show that every solution \( h \) of \( I_R \) satisfies the constraint \( R(a_0, b_0, c_0) \), i.e.
\[
h(a_0, b_0, c_0) \in R.
\]

According to the definition of \( n \)-torus, the positions of \( I_R \) split into two disjoint subsets, call them negative and positive, so that neighbors of a negative position are positive, and vice versa. Wlog. assume that \((a_0, b_0, c_0)\) is negative. Consider the following expression (symbol \( \prod \) stands for the group operation in \( R \), applied in an unspecified order):
\[
\prod_{(a,b,c) \text{ negative}} h(a, b, c)^{-1} \prod_{(a,b,c) \text{ positive}} h(a, b, c), \tag{16}
\]

where \((a, b, c)\) in the first expression ranges over all negative positions of \( I_R \) (hence \((a_0, b_0, c_0)\) is omitted), and in the second expression over all positive ones. The expression (16) evaluates to some value in \( R \); the value of the expression may however, in principle, depend on the order of application of the group operation. We aim at showing that irrespectively of the order of application of the group operation, the expression (16) evaluates to \( h(a_0, b_0, c_0) \), which implies (15).

Let \( f : R \to [R] \) be a surjective group homomorphism from \( R \) to a commutative group \([R]\), guaranteed jointly by Lemmas 2.7 and 2.8. Apply \( f \) to (16) to get an expression:
\[
\prod_{(a,b,c) \text{ negative}} f(h(a, b, c))^{-1} \prod_{(a,b,c) \text{ positive}} f(h(a, b, c)) \tag{17}
\]

which, irrespectively of the order of application of the group operation, evaluates to the same value in \([R]\). Observe that
\[
f(h(a)) \text{ appears in (17) exactly once, for every } a \in I \text{ different from } a_0, b_0, c_0; \text{ the same applies to the inverse } f(h(a))^{-1}.
\]

Thus, as \([R]\) is commutative, every \( f(h(a)) \) together with its inverse cancels out. Moreover, \( f(h(a_0)), f(h(b_0)) \) and \( f(h(c_0)) \) also appear in (17) exactly once, while their inverses do not appear as the negative position \((a_0, b_0, c_0)\) has been omitted. Thus the expression (17) evaluates to \( f(h(a_0, b_0, c_0)) \); as \( f \) is a group homomorphism we deduce that the expression (16) evaluates to \( h(a_0, b_0, c_0) \), as required. Proposition 5.5 is thus proved.

Acknowledgements. The author is grateful to Szymek Toruńczyk for proposing a simplified proof of Lemma 2.8. Moreover, thanks go to Bartek Klin, Ania Ochremiak, and Szymek Toruńczyk for long and fascinating discussions on CSP and its relationship to computation in sets with atoms. Finally, I thank Luc Segufoin for encouraging me to write down this note.

[1] T. Feder, M. Y. Vardi, The computational structure of monotone monadic SNP and constraint satisfaction: A study through Datalog and group theory, SIAM J. Comput. 28 (1) (1998) 57–104.
[2] T. Feder, Constraint satisfaction on finite groups with near subgroups, Electronic Colloquium on Computational Complexity (ECCC) (2005).
[3] A. A. Bulatov, V. Dalmu, A simple algorithm for Malcev constraints, SIAM J. Comput. 36 (1) (2006) 16–27.
[4] H. Ebbinghaus, J. Flum, Finite Model Theory, Perspectives in mathematical logic, Springer, 1999.
[5] V. Dalmau, B. Larose, Maltsev + Datalog \( \to \) symmetric Datalog, in: Proc. LICS, IEEE Computer Society, 2008, pp. 297–306.
[6] A. Atserias, A. A. Bulatov, A. Dawar, Axiom systems of equations and counting infinitary logic, Theor. Comput. Sci. 410 (18) (2009) 1666–1683.
[7] L. Barto, M. Kozik, Constraint satisfaction problems of bounded width, in: Proc. FOCS’09, IEEE Computer Society, 2009, pp. 595–603.
[8] L. Barto, M. Kozik, Constraint satisfaction problems solvable by local consistency methods, J. ACM 61 (1) (2014) 3.
[9] D. Hobby, R. McKenzie, The Structure of Finite Algebras, Vol. 76 of Contemporary Mathematics, AMS, Providence, R.I. 1988.
[10] B. Larose, L. Zádor, Bounded width problems and algebras, Algebra Universalis 56 (3-4) (2007) 39466.
[11] J. Cai, M. Furer, N. Immerson, An optimal lower bound on the number of variables for graph identifications, Combinatorica 12 (4) (1992) 389–410.
[12] M. Bojańczyk, B. Klin, S. Lasota, S. Toruńczyk, Turing machines with atoms, in: Proc. LICS’13, 2013, pp. 183–192.
[13] M. Bojańczyk, B. Klin, S. Lasota, Automata with group actions, in: Proc. LICS’11, 2011, pp. 355–364.
[14] L. Barto, The collapse of the bounded width hierarchy, Journal of Logic and Computation.
[15] G. L. Cherlin, G. A. Martin, D. H. Saracino, Arities of permutation groups: Wreath products and k-sets, Journal of Combinatorial Theory, Series A 74 (2) (1996) 249–286.