Quantified Constraints and Surjective Polymorphisms

Ferdinand Börner
Institut für Informatik
University of Potsdam
Potsdam, D-14482, Germany
fboerner@cs.uni-potsdam.de

Andrei Krokhin
Department of Computer Science
University of Warwick
Coventry, CV4 7AL, UK
andrei.krokhin@dcs.warwick.ac.uk

Andrei Bulatov, Peter Jeavons
Oxford University Computing Laboratory
Oxford, OX1 3QD, UK
{andrei.bulatov,peter.jeavons}@comlab.ox.ac.uk

December 22, 2002

Abstract

It is well known that the complexity of a constraint satisfaction problem parameterized by the set of allowed constraint predicates is determined by the polymorphisms of those predicates. In this paper we consider a more general framework for constraint satisfaction problems which allows arbitrary quantifiers over constrained variables, rather than just existential quantifiers. We show that the complexity of such extended problems is determined by the surjective polymorphisms of the constraint predicates. We give examples to illustrate how this result can be used to identify tractable and intractable cases. Finally, we apply the technique presented here to completely classify the complexity of such quantified constraint satisfaction problems when the parameter set contains all graphs of permutations.

1 Introduction

The constraint satisfaction problem (CSP) provides a general framework in which a wide variety of combinatorial problems can be expressed in a natural way [10, 35]. A constraint satisfaction problem instance can be viewed as a collection of predicates on overlapping sets of variables. The aim is to determine whether there exist values for all of the variables such that all of the specified predicates hold simultaneously. The standard constraint satisfaction problem can be parameterised by restricting the set of allowed predicates which can be used as constraints. The problem of determining (up to complete classification) the complexity of the CSP (and its many variants) for all possible parameter sets has attracted much attention, partly because constraint satisfaction problems play an important role in Artificial Intelligence [35], and partly because they "present a reasonably accurate bird's-eye view on complexity theory" [10]. One important outcome of research in this direction has been the design of sophisticated new polynomial-time algorithms for solving a wide variety of problems (see, e.g., [4, 13]). Another outcome has been progress with some important issues in complexity theory, such as the discovery of large subclasses of complexity classes that avoid intermediate complexity (e.g., dichotomy results in the case of **NP**, see [3, 15, 34]).

For the Boolean (i.e., two-valued) case, the complexity of the standard constraint satisfaction problem has been studied from the above perspective [34], as well as a number of related problems, including quantified and counting problems, maximum (and minimum) satisfiability, generating all solutions, optimizing the number of positive truth values in a solution (all of these are in [10]), minimal satisfiability [24], unique satisfiability [21], deciding equivalence and isomorphism of instances [2], maximizing Hamming distance between solutions [11], finding lexicographically minimal (or maximal) solutions [32], inverse satisfiability [22], and solving random instances [9].

Analysing the complexity of non-Boolean CSPs is a significantly more difficult task: these problems usually withstand a direct combinatorial approach, and so require more involved techniques. A far-reaching approach via graph theory, logic and games has been developed in [14, 15, 26]. However, the most successful approach so far has been the algebraic approach developed in [6, 19, 20]. This approach has led to a number of new results (see, e.g., [4, 5, 7, 13, 14]), and has culminated (so far) in a complete classification of the complexity of parameterized CSPs for the three-valued case [3].

The standard CSP can be expressed as follows: given a first-order sentence consisting of a conjunction of predicates, where all of the variables are existentially quantified, determine whether the sentence is true. One of the most natural generalisations of this framework is to consider the *quantified constraint satisfaction problem* (QCSP), in which universal quantifiers are allowed in the sentence, as well as existential quantifiers [10, 12]. This generalisation greatly increases the expressive power of the framework, but also increases the complexity of deciding whether an arbitrary instance is true — from **NP**-complete to **PSPACE**-complete.

Boolean QCSP (also known as QSAT or QBF) and some of its restrictions (such as Q3SAT) have always been standard examples of **PSPACE**-complete problems [17, 28, 34]. For some parameter sets, Boolean QCSP has been shown to be tractable: for all binary predicates in [1], and for Horn predicates in [25]. Finally, a complete classification for the Boolean case was obtained in [10, 12]. However, to the best of our knowledge, only trivial results are known about the complexity of non-Boolean QCSPs.

In this paper we extend the algebraic approach for the first time to the more general framework of the quantified constraint satisfaction problem over an arbitrary finite set of values. We show that certain algebraic objects (surjective polymorphisms) determine the complexity of the QCSP for any given value of the parameter. We then give examples to show how this result can be used to identify quantified constraint satisfaction problems lying in more restricted complexity classes such as **NL** and **PTIME**. Finally, we obtain the first complete classification result for a class of general (non-Boolean) quantified constraint satisfaction problems.

2 Preliminaries

Throughout the paper we use the standard correspondence between predicates and relations: a relation consists of all tuples of values for which the corresponding predicate holds. We will use the same symbol for a predicate and its corresponding relation, since the meaning will always be clear from the context. We will use $R_D^{(m)}$ to denote the set of all m-ary relations (or predicates) over a set D, and R_D to denote the set $\bigcup_{m=1}^{\infty} R_D^{(m)}$.

Definition 2.1 *Let $\Gamma \subseteq R_D$. An* instance *of* $\mathrm{CSP}(\Gamma)$ *is a first-order sentence $\exists x_1 \ldots \exists x_l \; (\varrho_1 \wedge \ldots \wedge \varrho_q)$, where each ϱ_i is an atomic formula involving a predicate from Γ, and $x_1, \ldots, x_l$ are the variables appearing in the ϱ_i. The* question *is whether the sentence is true.*

The predicates ϱ_i appearing in an instance will be referred to as *constraints*, since each of them restricts the possible models for the instance in some way.

In addition to predicates and relations we will also consider arbitrary *operations* on the set of values. We will use $O_D^{(n)}$ to denote the set of all n-ary operations on a set D (that is, the set of mappings $f\colon D^n \to D$), and O_D to denote the set $\bigcup_{n=1}^{\infty} O_D^{(n)}$.

Any operation on D can be extended in a standard way to an operation on tuples over D, as follows. For any operation $f \in O_D^{(n)}$, and any collection of tuples $\vec{a}_1, \vec{a}_2, \ldots, \vec{a}_n \in D^m$, where $\vec{a}_i = (\vec{a}_i(1), \ldots, \vec{a}_i(m))$ $(i = 1 \ldots n)$, define $f(\vec{a}_1, \ldots, \vec{a}_n) = (\, f(\vec{a}_1(1), \ldots, \vec{a}_n(1)), \ldots, f(\vec{a}_1(m), \ldots, \vec{a}_n(m))\,)$.

Definition 2.2 *For any relation $\varrho \in R_D^{(m)}$, and any operation $f \in O_D^{(n)}$, if $f(\vec{a}_1, \ldots, \vec{a}_n) \in \varrho$ for all choices of $\vec{a}_1, \ldots, \vec{a}_n \in \varrho$, then ϱ is said to be* invariant *under f, and f is called a* polymorphism *of ϱ.*

The set of all relations which are invariant under each operation from some set $C \subseteq O_D$ will be denoted $\mathsf{Inv}(C)$. The set of all operations which are polymorphisms of every relation from some set $\Gamma \subseteq R_D$ will be denoted $\mathsf{Pol}(\Gamma)$. The operators Inv and Pol form a Galois correspondence between R_D and O_D. A basic introduction to this correspondence can be found in [29], and a comprehensive study in [30].

The following result was obtained in [19].

Theorem 2.3 *Let Γ_1 and Γ_2 be sets of predicates over a finite set, such that Γ_1 is finite. If $\mathsf{Pol}(\Gamma_2) \subseteq \mathsf{Pol}(\Gamma_1)$ then $\mathrm{CSP}(\Gamma_1)$ is polynomial-time reducible to $\mathrm{CSP}(\Gamma_2)$.*

This result shows that, when the set of values is finite, finite sets of relations with the same polymorphisms give rise to constraint satisfaction problems which are mutually reducible. In other words, the complexity of $\mathrm{CSP}(\Gamma)$ is determined by the polymorphisms of Γ.

A number of results on the complexity of constraint satisfaction problems have been obtained via this approach (e.g., [3, 4, 5, 6, 7, 13, 14, 19, 20]). For example, Schaefer's Dichotomy Theorem [34], when appropriately re-stated, easily follows from Theorem 2.3 and well-known algebraic results [31] (see [19]).

Theorem 2.4 *For any $\Gamma \subseteq R_{\{0,1\}}$, $\mathrm{CSP}(\Gamma)$ is in* **PTIME** *when $\mathsf{Pol}(\Gamma)$ contains one of the following:*

- *the constant 0 or constant 1 operations,*
- *the conjunction or disjunction operations,*
- *the affine operation $x - y + z \pmod 2$,*
- *the majority operation $(x \vee y) \wedge (x \vee z) \wedge (y \vee z)$.*

In all other cases $\mathrm{CSP}(\Gamma)$ is **NP***-complete.*

In this paper we consider the more general framework of the *quantified constraint satisfaction problem*, which is defined as follows.

Definition 2.5 *Let $\Gamma \subseteq R_D$. An* instance *of $\mathrm{QCSP}(\Gamma)$ is a first-order sentence $\mathcal{Q}_1 x_1 \ldots \mathcal{Q}_l x_l \; (\varrho_1 \wedge \ldots \wedge \varrho_q)$, where each ϱ_i is an atomic formula involving a predicate from Γ, $x_1, \ldots, x_l$ are the variables appearing in the ϱ_i, and $\mathcal{Q}_1, \ldots, \mathcal{Q}_l$ are arbitrary quantifiers. The* question *is whether the sentence is true.*

Clearly, an instance of $\mathrm{CSP}(\Gamma)$ corresponds to an instance of $\mathrm{QCSP}(\Gamma)$ in which all the quantifiers happen to be existential.

For any finite set D, and any set of relations $\Gamma \subseteq R_D$, one can use an exhaustive search algorithm to show that $\mathrm{QCSP}(\Gamma)$ is in **PSPACE**. The complexity of $\mathrm{QCSP}(\Gamma)$ has been completely characterized in the Boolean case [10, 12].

Theorem 2.6 *For any $\Gamma \subseteq R_{\{0,1\}}$, $\mathrm{QCSP}(\Gamma)$ is in* **PTIME** *when $\mathsf{Pol}(\Gamma)$ contains one of the following:*

- *the conjunction or disjunction operations,*
- *the affine operation $x - y + z \pmod 2$,*
- *the majority operation $(x \vee y) \wedge (x \vee z) \wedge (y \vee z)$.*

In all other cases QCSP(Γ) *is* **PSPACE**-*complete.*

Corollary 2.7 QCSP($\{N\}$) *is* **PSPACE**-*complete, where N is the ternary "not-all-equal" relation on $\{0,1\}$, defined by $N = \{0,1\}^3 \setminus \{(0,0,0),(1,1,1)\}$*

Even though Theorem 2.6 is stated using polymorphisms, it was proved using a combinatorial rather than an algebraic approach, and this method of proof does not easily generalise to larger sets of values. In the remaining sections we introduce an algebraic approach which enables us to systematically analyse the complexity of quantified constraint satisfaction problems over an arbitrary finite set D.

3 Reduction in QCSP

The main result of this paper shows that, for quantified constraint satisfaction problems, *surjective* polymorphisms play a similar role to that played by arbitrary polymorphisms for ordinary CSPs (cf. Theorem 2.3). Let $\mathsf{s\text{-}Pol}(\Gamma)$ denote the set of all surjective operations from $\mathsf{Pol}(\Gamma)$.

Theorem 3.1 *Let Γ_1 and Γ_2 be sets of predicates over a finite set, such that Γ_1 is finite. If $\mathsf{s\text{-}Pol}(\Gamma_2) \subseteq \mathsf{s\text{-}Pol}(\Gamma_1)$, then* QCSP($\Gamma_1$) *is polynomial-time reducible to* QCSP(Γ_2).

This theorem follows immediately from the next two propositions whose proofs can be found in the Appendix.

Definition 3.2 *For any set $\Gamma \subseteq R_D$, the set $[\Gamma]$ consists of all predicates that can be expressed using*

1. *predicates from Γ, together with the binary equality predicate on D (denoted $=_D$),*
2. *conjunction,*
3. *existential quantification,*
4. *universal quantification.*

Proposition 3.3 *Let Γ_1 and Γ_2 be sets of predicates over a finite set, such that Γ_1 is finite. If $[\Gamma_1] \subseteq [\Gamma_2]$, then* QCSP($\Gamma_1$) *is polynomial-time reducible to* QCSP(Γ_2).

Proposition 3.4 *For any set of predicates Γ over a finite set, $[\Gamma] = \mathsf{Inv}(\mathsf{s\text{-}Pol}(\Gamma))$.*

4 Intractable Cases

In this section we will use Theorem 3.1 to show that certain small sets of relations give rise to quantified constraint satisfaction problems that are intractable. In particular, we will show that QCSP(Γ) can be **PSPACE**-complete even in some cases where CSP(Γ) is trivial.

We first establish that a particular QCSP problem is **PSPACE**-complete. This problem corresponds to a generalised form of the standard GRAPH-$|D|$-COLORABILITY problem (which can be expressed as CSP($\{\neq_D\}$) where $\neq_D$ is the binary disequality predicate on D) [17, 28],

Proposition 4.1 QCSP($\{\neq_D\}$) *is* **PSPACE**-*complete when $|D| \geq 3$.*

Proof: By reduction from QCSP($\{N\}$), where N is the ternary not-all-equal predicate on a 2-valued set, as defined in Corollary 2.7. For details see Appendix. ∎

Theorem 4.2 *For any finite set D with $|D| \geq 3$, and any $\Gamma \subseteq R_D$, if every $f \in \mathsf{s\text{-}Pol}(\Gamma)$ is of the form $f(x_1, \ldots, x_n) = g(x_i)$ for some $1 \leq i \leq n$ and some permutation g on D, then* $\mathrm{QCSP}(\Gamma)$ *is* **PSPACE**-*complete.*

Proof: By Lemma 1.3.1 (b) of [30], $\mathsf{Pol}(\{\neq_D\})$, for $|D| \geq 3$, consists of all functions of the form described in the Theorem. Hence $\mathsf{Pol}(\{\neq_D\}) = \mathsf{s\text{-}Pol}(\{\neq_D\})$, and we can apply Theorem 3.1 and Proposition 4.1. ∎

The next example uses this result to show that even predicates that give rise to trivial constraint satisfaction problems can give rise to intractable quantified constraint satisfaction problems. This can happen because non-surjective operations, which may guarantee the tractability of the CSP, do not affect the complexity of the QCSP.

Example 4.3 *Let τ_s be the s-ary "not-all-distinct" predicate holding on a tuple $(a_1, \ldots, a_s)$ if and only if $|\{a_1, \ldots, a_s\}| < s$. Note that $\tau_s \supseteq \{(a, \ldots, a) \mid a \in D\}$, so every instance of* $\mathrm{CSP}(\{\tau_s\})$ *is trivially satisfiable by assigning the same value to all variables.*

However, by Lemma 2.2.4 of [30], the set $\mathsf{Pol}(\{\tau_{|D|}\})$ contains all non-surjective operations on D, together with all operations of the form given in Theorem 4.2. Hence, $\{\tau_{|D|}\}$ satisfies the conditions of Theorem 4.2, and $\mathrm{QCSP}(\{\tau_{|D|}\})$ *is* **PSPACE**-*complete. Similar arguments can be used to show that* $\mathrm{QCSP}(\{\tau_s\})$ *is* **PSPACE**-*complete, for any s in the range $3 \leq s \leq |D|$.*

5 Tractable cases

In spite of the results of the previous section, it is possible to identify sets of predicates which give rise to tractable QCSP problems. In this section we identify and describe two families of predicates of this kind.

5.1 Affine predicates

For an Abelian group G the operation $f(x, y, z) = x - y + z$ is called the *affine operation* of G. Relations invariant under the affine operation of a finite Abelian group play a significant role in the study of the complexity of the standard constraint satisfaction problem [4, 15, 19, 20]. We are going to generalise and extend these earlier results by showing that the problem $\mathrm{QCSP}(\Gamma)$ is tractable for any set of predicates which is invariant under the affine operation of a finite Abelian group.

Theorem 5.1 *Let G be a finite Abelian group, with affine operation f. The problem class* $\mathrm{QCSP}(\mathsf{Inv}(\{f\}))$ *is in* **PTIME**.

This result follows immediately from the results later in this section.

Throughout this section, let G be a finite Abelian group, with affine operation f, and unit element 0, and let Γ be a set of relations over G which is invariant under f.

Our strategy is to reduce problems from $\mathrm{QCSP}(\Gamma)$ to solving systems of linear equations of a rather general kind. To this end we first note that, by straightforward algebraic manipulation, it can be shown that any (n-ary) relation invariant under f is a coset of a subgroup of the group G^n. We will show below that every such coset is representable as the set of solutions to a system of linear equations.

In the simplest case, when the order of G is prime, G can be considered as a prime field, and hence G^n as a vector space over G. In this case, each coset of a subgroup of G^n is a hyperplane, and it is well-known that such hyperplanes can be defined by systems of linear equations, whose coefficients are elements of the field G. In the more general case, for an arbitrary Abelian group, we will obtain a similar characterisation of the cosets in terms of certain kinds of linear equations, but in this case we must allow the coefficients of these linear equations to be arbitrary *endomorphisms* of G, which are defined as follows.

Definition 5.2 *Given any group G, a mapping $\varphi: G \to G$ is said to be an* endomorphism *of G if $\varphi(g+h) = \varphi(g) + \varphi(h)$ for any $g, h \in G$.*

To simplify the notation we shall denote endomorphisms by the letter r, with subscripts, and we will write rg instead of $r(g)$.

The next proposition allows us to express any predicate in Γ as the set of solutions to a system of linear equations with coefficients in $\mathsf{End}\ G$. We first define some notation. For any subgroup H of G^n, define

$$H^\perp = \{(r_1, \ldots, r_n) \in (\mathsf{End}\ G)^n \mid r_1 a_1 + \ldots + r_n a_n = 0 \text{ for all } (a_1, \ldots, a_n) \in H\}.$$

Similarly, for any subset R of $(\mathsf{End}\ G)^n$ define

$$R^\perp = \{(a_1, \ldots, a_n) \in G^n \mid r_1 a_1 + \ldots + r_n a_n = 0 \text{ for all } (r_1, \ldots, r_n) \in R\}.$$

Proposition 5.3 *For any finite Abelian group G, if H is a subgroup of G^n, then $H^{\perp\perp} = H$.*

Proof: See Appendix. ∎

Corollary 5.4 *For every coset ϱ of a subgroup of G^n, there is a linear system of equations L with coefficients in $\mathsf{End}\ G$, such that a tuple $\vec{a} \in G^n$ belongs to ϱ if and only if $\vec{a}$ is a solution to L. Moreover, L can be constructed in polynomial time and contains only polynomially many equations and unknowns.*

Proof: See Appendix. ∎

It follows from Corollary 5.4 that, given any problem instance $\mathcal{P}$ in QCSP(Γ), where

$$\mathcal{P} = \mathcal{Q}_1 x_1 \ldots \mathcal{Q}_l x_l \quad (\varrho_1 \wedge \ldots \wedge \varrho_q),$$

we can represent each ϱ_i by a linear system of equations L_i with coefficients in $\mathsf{End}\ G$. Hence, $\mathcal{P}$ is equivalent to

$$\mathcal{Q}_1 x_1 \ldots \mathcal{Q}_l x_l\ \mathcal{L}$$

where $\mathcal{L}$ consists of the union of all equations from $L_1, \ldots, L_q$.

Finally, the next result shows that sentences of this form can be decided in polynomial time.

Proposition 5.5 *For any finite Abelian group G, there is a polynomial time algorithm that decides any sentence of the form $\mathcal{Q}_1 x_1 \ldots \mathcal{Q}_l x_l\ \mathcal{L}$, where $\mathcal{L}$ is a linear system of equations with coefficients in $\mathsf{End}\ G$.*

Proof: See Appendix. ∎

5.2 Implicative predicates

Our second example of predicates which give rise to tractable quantified constraint satisfaction problems concerns predicates which are invariant under an operation known as the *dual discriminator*. These problems can be viewed as generalized Q2SAT, and our algorithm generalises and extends the algorithm for Q2SAT given in [1].

Definition 5.6 *For any finite set D, the* dual discriminator *operation d on D, is given by*

$$d(x, y, z) = \begin{cases} y & \text{if } y = z \\ x & \text{otherwise} \end{cases}$$

Note that, in the special case when $D = \{0, 1\}$, the dual discriminator is exactly the majority operation mentioned in Theorems 2.4 and 2.6.

Theorem 5.7 *Let D be a finite set, and let d be the dual discriminator operation on D. The problem class* $\mathrm{QCSP}(\mathsf{Inv}(\{d\}))$ *is in* **NL**.

To establish this result, we consider a graph structure associated with any instance of a constraint satisfaction problem, which is sometimes known as the "microstructure graph" [16]. For any instance $\mathcal{P}$ of QCSP, with variables V and set of values D, the *microstructure graph* of $\mathcal{P}$ is the graph (N, E), where the set of nodes, N, is the subset of pairs $V \times D$ representing all possible assignments of values to individual variables which are compatible with the constraints on those variables, and the set of edges, E, is the set of all assignments to pairs of variables which are compatible with the constraints on those pairs. In other words, E is the set of all (ordered) pairs of nodes $((v_i, a), (v_j, b))$ such that $v_i \neq v_j$ and the partial assignment $v_i = a, v_j = b$ is compatible with each individual predicate in $\mathcal{P}$.

We will call an arc $((v_i, a), (v_j, b))$ in the microstructure graph of $\mathcal{P}$ *implicative* if it is the unique arc in E with first component (v_i, a) and second component (v_j, b'), for some value b'. If we remove all arcs from the microstructure graph of $\mathcal{P}$ which are *not* implicative, then we obtain a directed graph which will be called the *implication graph* of $\mathcal{P}$, and denoted $\mathrm{I}(\mathcal{P})$.

Note that the arcs in the implication graph $\mathrm{I}(\mathcal{P})$ represent logical implications which can be made about the possible models of $\mathcal{P}$. If any model gives value a to variable v_i, and $\mathrm{I}(\mathcal{P})$ contains the arc $((v_i, a), (v_j, b))$, then that model must also give the value b to the variable v_j. By the transitivity of implication, we obtain the same conclusion if there is any path from (v_i, a) to (v_j, b) in $\mathrm{I}(\mathcal{P})$.

For some sets of predicates the implication graph has certain additional properties, which can be used to obtain further restrictions on the possible models. We define one such property as follows.

Definition 5.8 *For any QCSP instance $\mathcal{P}$, the implication graph $\mathrm{I}(\mathcal{P})$ will be called* invertible *if whenever there is a path from a node (v_i, a) to a node (v_j, b), then for each node (v_j, b') with $b' \neq b$ there is a path from (v_j, b') to (v_i, a') for some $a' \neq a$ (which may depend on b').*

The second property we define holds when the implication graph captures all the restrictions on the possible models. In other words, it says that a given assignment is a model whenever it satisfies all of the implications encoded in the implication graph.

Definition 5.9 *For any QCSP instance $\mathcal{P}$, the implication graph $\mathrm{I}(\mathcal{P})$ will be called* sufficient *if a given assignment σ is a model of $\mathcal{P}$ whenever $\mathrm{I}(\mathcal{P})$ contains the node $(v, \sigma(v))$ for each variable v, and does not contain an arc $((v_i, \sigma(v_i)), (v_j, b))$, with $b \neq \sigma(v_j)$.*

We will now show that, when the implication graph for an instance is both invertible and sufficient, then there is a non-deterministic logarithmic-space algorithm for that instance.

Proposition 5.10 *There is a non-deterministic logarithmic-space algorithm that decides any QCSP instance $\mathcal{P}$ whose implication graph $\mathrm{I}(\mathcal{P})$ is both invertible and sufficient.*

Proof: See Appendix. ∎

Finally, we show that both of these properties hold for all predicates that are invariant under the dual discriminator.

Proposition 5.11 *Let D be a finite set, and let d be the dual discriminator operation on D. For any instance $\mathcal{P}$ in* $\mathrm{QCSP}(\mathsf{Inv}(\{d\}))$, *the implication graph $\mathrm{I}(\mathcal{P})$ is invertible and sufficient.*

Proof: It was shown in [20] that the predicates invariant under d are precisely the predicates which can be expressed as conjunctions of binary predicates, each of which is of the form of a "0/1/all" binary relation, as described in [8]. Such predicates are also described in [23], where they are referred to as "implicational".

By examining the structure of "0/1/all" relations, as described in [8], it is straightforward to show that they give rise to implication graphs which are invertible. (In fact, an equivalent statement appears as Proposition 2.3 in [23].)

Furthermore, by examining the structure of "0/1/all" relations, it is straightforward to verify that in any problem instance involving such relations, the only restrictions imposed on the models are those captured by the implication graph of the instance. Hence this implication graph is sufficient. ∎

6 A Trichotomy Result

In this section we apply results from the previous sections to obtain a complete classification of complexity of QCSP(Γ) in those cases where Γ contains the set Δ of all graphs of permutations. Recall that the graph of a permutation π is the binary relation $\{(x,y) \mid y = \pi(x)\}$ (or the binary predicate $\pi(x) = y$). The complexity of CSP(Γ) for such sets Γ is completely classified in [13].

We will need two new surjective operations:

- The k-ary *near projection* operation, $l_k(x_1,\ldots,x_k) = \begin{cases} x_1 & \text{if } x_1,\ldots,x_k \text{ are all different,} \\ x_k & \text{otherwise.} \end{cases}$
- The ternary *switching* operation, $s(x,y,z) = \begin{cases} x & \text{if } y = z, \\ y & \text{if } x = z, \\ z & \text{otherwise.} \end{cases}$

Proposition 6.1 *If* $\Gamma \subseteq R_D$, $|D| \geq 3$, *and* $l_{|D|} \in \mathsf{s\text{-}Pol}(\Gamma)$ *then* QCSP(Γ) *is polynomial-time reducible to* CSP(Γ') *where* $\Gamma' = \mathsf{Inv}(\mathsf{Pol}(\Gamma))$. *In particular,* QCSP($\Gamma$) *is in* **NP**.

Proof: See Appendix. ∎

Theorem 6.2 *Let* $\Delta \subseteq \Gamma \subseteq R_D$, *and* $|D| \geq 3$.

- *If* $\mathsf{s\text{-}Pol}(\Gamma)$ *contains the dual discriminator* d, *or the switching operation* s, *or an affine operation, then* QCSP(Γ) *is in* **PTIME**;
- *else, if* $\mathsf{s\text{-}Pol}(\Gamma)$ *contains* $l_{|D|}$, *then* QCSP(Γ) *is* **NP**-*complete;*
- *else* QCSP(Γ) *is* **PSPACE**-*complete.*

Proof: It follows from [27] that, either $\mathsf{s\text{-}Pol}(\Gamma)$ consists of all projections (that is, of functions of the form described in Theorem 4.2 with g being the identity function), or this set contains d, or $l_{|D|}$, or (when $|D| \in \{3,4\}$) an affine operation. If $\mathsf{s\text{-}Pol}(\Gamma)$ consists of all projections then, by Theorem 4.2, QCSP(Γ) is **PSPACE**-complete. If $\mathsf{s\text{-}Pol}(\Gamma)$ contains the dual discriminator or an affine operation then, by Theorem 5.1 or Theorem 5.7, QCSP(Γ) is tractable. Suppose that $\mathsf{s\text{-}Pol}(\Gamma)$ contains $l_{|D|}$. Then, by Proposition 6.1, this problem is polynomial-time reducible to CSP(Γ'). If $\mathsf{s\text{-}Pol}(\Gamma)$ contains s or d then clearly so does $\mathsf{Pol}(\Gamma')$. It follows from the results of [13] that in this case CSP(Γ') is tractable, and hence so is QCSP(Γ). If $\mathsf{s\text{-}Pol}(\Gamma)$ contains neither s nor d then, by [13], CSP(Γ) is **NP**-complete. Since, obviously, CSP(Γ) is polynomial-time reducible to QCSP(Γ), and QCSP(Γ) is in **NP**, the result follows. ∎

Note that all conditions in Theorem 6.2 can be efficiently checked. Moreover, the theorem remains true when the set Δ is replaced by any collection of graphs of permutations such that these permutations generate the full symmetric group on D.

7 Conclusion

We have shown that the algebraic theory relating complexity and polymorphisms, which was originally developed for the standard constraint satisfaction problem allowing only existential quantifiers, can be extended to deal with the more general framework of the quantified constraint satisfaction problem.

In this extension of the theory it turns out that it is the *surjective* polymorphisms of the predicates used in problem instances which determine the complexity of the corresponding problems. Using this information we have been able to identify subproblems of the quantified constraint satisfaction problem lying in (or complete for) some standard complexity classes, and to obtain a complete classification of complexity for certain special cases. We expect that by developing the results and ideas from this paper one will be able to successfully build a far-reaching theory linking algebraic properties of relations with the computational complexity of certain associated problems, just as it was in the case of ordinary constraint satisfaction problems.

References

[1] B. Aspvall, M.F. Plass, and R.E. Tarjan. A linear time algorithm for testing the truth of certain quantified Boolean formulas. *Information Processing Letters*, 8:121–123, 1979.

[2] E. Böhler, E. Hemaspaandra, S. Reith, and H. Vollmer. Equivalence and isomorphism for Boolean constraint satisfaction. In *Proceedings 16th International Workshop on Computer Science Logic, CSL'02*, volume 2471 of *Lecture Notes in Computer Science*, pages 412–426. Springer-Verlag, 2002.

[3] A.A. Bulatov. A dichotomy theorem for constraints on a three-element set. In *Proceedings 43rd IEEE Symposium on Foundations of Computer Science, FOCS'02*, 2002. to appear.

[4] A.A. Bulatov. Mal'tsev constraints are tractable. Technical Report RR-02-05, Computing Laboratory, Oxford University, 2002.

[5] A.A. Bulatov and P.G. Jeavons. Algebraic structures in combinatorial problems. Technical Report MATH-AL-4-2001, Technische Universität Dresden, 2001.

[6] A.A. Bulatov, A.A. Krokhin, and P.G. Jeavons. Constraint satisfaction problems and finite algebras. In *Proceedings 27th International Colloquium on Automata, Languages and Programming, ICALP'00*, volume 1853 of *Lecture Notes in Computer Science*, pages 272–282. Springer-Verlag, 2000.

[7] A.A. Bulatov, A.A. Krokhin, and P.G. Jeavons. The complexity of maximal constraint languages. In *Proceedings 33rd ACM Symposium on Theory of Computing, STOC'01*, pages 667–674, 2001.

[8] M.C. Cooper, D.A. Cohen, and P.G. Jeavons. Characterising tractable constraints. *Artificial Intelligence*, 65:347–361, 1994.

[9] N. Creignou and H. Daudé. Random generalized satisfiability problems. In *Proceedings of 5th International Symposium on Theory and Applications of Satisfiability Testing - SAT'02*, pages 17–26, 2002.

[10] N. Creignou, S. Khanna, and M. Sudan. *Complexity Classification of Boolean Constraint Satisfaction Problems*, volume 7 of *SIAM Monographs on Discrete Mathematics and Applications*. 2001.

[11] P. Crescenzi and G. Rossi. On the Hamming distance of constraint satisfaction problems. *Theoretical Computer Science*, 288(1):85–100, 2002.

[12] V. Dalmau. Some dichotomy theorems on constant-free quantified Boolean formulas. Technical Report TR LSI-97-43-R, Department LSI, Universitat Politecnica de Catalunya, 1997.

[13] V. Dalmau. A new tractable class of constraint satisfaction problems. In *Proceedings 6th International Symposium on Artificial Intelligence and Mathematics*, 2000.

[14] V. Dalmau. Constraint satisfaction problems in non-deterministic logarithmic space. In *Proceedings 29th International Colloquium on Automata, Languages and Programming, ICALP'02*, volume 2380 of *Lecture Notes in Computer Science*, pages 414–425. Springer-Verlag, 2002.

[15] T. Feder and M.Y. Vardi. The computational structure of monotone monadic SNP and constraint satisfaction: A study through Datalog and group theory. *SIAM Journal of Computing*, 28:57–104, 1998.

[16] E.C. Freuder. Exploiting structure in constraint satisfaction problems. In M. Mayoh, E. Tyugum, and J. Penjam, editors, *Constraint Programming*, volume 131 of *NATO ASI Series*. Springer-Verlag, 1993.

[17] M. Garey and D.S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. Freeman, San Francisco, CA., 1979.

[18] T.H. Jackson. *Number Theory*. Routledge and Kegan Paul, 1975.

[19] P.G. Jeavons. On the algebraic structure of combinatorial problems. *Theoretical Computer Science*, 200:185–204, 1998.

[20] P.G. Jeavons, D.A. Cohen, and M. Gyssens. Closure properties of constraints. *Journal of the ACM*, 44:527–548, 1997.

[21] L. Juban. Dichotomy theorem for generalized unique satisfiability problem. In *Proceedings 12th Conference on Fundamentals of Computation Theory, FCT'99*, volume 1684 of *Lecture Notes in Computer Science*, pages 327–337. Springer-Verlag, 1999.

[22] D. Kavvadias and M. Sireni. The inverse satisfiability problem. *SIAM Journal on Computing*, 28(1):152–163, 1998.

[23] L. Kirousis. Fast parallel constraint satisfaction. *Artificial Intelligence*, 64:147–160, 1993.

[24] L. Kirousis and Ph. Kolaitis. The complexity of minimal satisfiability problems. In *Proceedings 18th International Symposium on Theoretical Aspects of Computer Science, STACS'01*, volume 2010 of *Lecture Notes in Computer Science*, pages 407–418. Springer-Verlag, 2001.

[25] H. Kleine Büning, M. Karpinski, and A. Flögel. Resolution for quantified Boolean formulas. *Information and Computation*, 117(1):12–18, 1995.

[26] Ph.G. Kolaitis and M.Y. Vardi. Conjunctive-query containment and constraint satisfaction. *Journal of Computer and System Sciences*, 61:302–332, 2000.

[27] S.S. Marchenkov. Homogeneous algebras. *Problemy Kibernetiki*, 39:85–106, 1982. (Russian).

[28] C.H. Papadimitriou. *Computational Complexity*. Addison-Wesley, 1994.

[29] N. Pippenger. *Theories of Computability*. Cambridge University Press, Cambridge, 1997.

[30] R. Pöschel and L.A. Kalužnin. *Funktionen- und Relationenalgebren*. DVW, Berlin, 1979.

[31] E.L. Post. *The two-valued iterative systems of mathematical logic*, volume 5 of *Annals Mathematical Studies*. Princeton University Press, 1941.

[32] S. Reith and H. Vollmer. Optimal satisfiability for propositional calculi and constraint satisfaction problems. In *Proceedings 25th International Symposium on Mathematical Foundations of Computer Science, MFCS'00*, volume 1893 of *Lecture Notes in Computer Science*, pages 640–649. Springer, 2000.

[33] J.J. Rotman. *An Introduction to the Theory of Groups*. Graduate Texts in Mathematics 148. Springer Verlag, 4th edition, 1995.

[34] T.J. Schaefer. The complexity of satisfiability problems. In *Proceedings 10th ACM Symposium on Theory of Computing, STOC'78*, pages 216–226, 1978.

[35] E. Tsang. *Foundations of Constraint Satisfaction*. Academic Press, London, 1993.

A Appendix

A.1 Proof of Proposition 3.3

Let D be a finite set, and let $\Gamma_1, \Gamma_2 \subseteq R_D$. By Definition 3.2, for any n-ary relation ϱ in $[\Gamma_2]$, the predicate $\varrho(x_1, \ldots, x_n)$ is equivalent to a formula Φ_ϱ of the form $\mathcal{Q}_1 y_1 \ldots \mathcal{Q}_m y_m \mathcal{C}$, where the $\mathcal{Q}_i$, $1 \le i \le m$ are quantifiers, and $\mathcal{C}$ is a conjunction of atomic formulas involving only predicates from $\Gamma_2 \cup \{=_D\}$ and variables $x_1, \ldots, x_n, y_1, \ldots, y_m$.

Let sentence $\mathcal{P}_0$ be an instance of $\mathrm{QCSP}(\Gamma_1)$. Replace each predicate ϱ in $\mathcal{P}_0$ by the corresponding formula Φ_ϱ, to obtain an equivalent formula $\mathcal{P}_1$. Transform $\mathcal{P}_1$ into prenex normal form by moving all quantifiers (in order) to the front of the formula (renaming variables as needed to avoid name clashes). This transformation can be carried out in logarithmic space. The resulting sentence, $\mathcal{P}_2$, is an instance of $\mathrm{QCSP}(\Gamma_2 \cup \{=_D\})$, and $\mathcal{P}_2$ is clearly equivalent to $\mathcal{P}_0$.

It now only remains to remove any occurrences of the equality relation from $\mathcal{P}_2$. We shall assume that $|D| \ge 2$ (the case $|D| = 1$ is trivial). Consider the graph $G = (V, E)$ whose vertices are the variables appearing in $\mathcal{P}_2$ and

$$E = \{(x, y) \in V^2 \mid (x = y) \text{ is a subformula in } \mathcal{P}_2\}.$$

Compute the connected components of G. For each connected component K, if K contains two variables, x and y, such that y is universally quantified (in $\mathcal{P}_2$), and x is free within the scope of y, then $\mathcal{P}_2$ (and hence $\mathcal{P}_0$) is obviously false. In the remaining cases, if the variables in K appear in $\mathcal{P}_2$ only in equalities, then simply remove these equalities from $\mathcal{P}_2$ along with the corresponding quantifiers. Otherwise, let x be a variable in K that is free, within the scope of all other variables in K. Remove from $\mathcal{P}_2$ the quantifiers corresponding to $K \setminus \{x\}$ and replace all variables from K with x.

After processing all connected components of G in this way we obtain a sentence $\mathcal{P}_3$ that is equivalent to $\mathcal{P}_0$ and is an instance of $\mathrm{QCSP}(\Gamma_2)$. It is easy to check that, since Γ_1 is finite, the whole transformation can be carried out in polynomial time.

A.2 Proof of Proposition 3.4

Let D be a finite set, and let $\Gamma \subseteq R_D$. The equality relation, $=_D$, is invariant under every operation on D, so $\Gamma \cup \{=_D\} \subseteq \mathsf{Inv}(\mathsf{s\text{-}Pol}(\Gamma))$. Let f be a surjective operation on D. It is straightforward to verify that applying conjunction or any quantification to predicates invariant under f gives another predicate which is also invariant under f. Hence, $[\Gamma] \subseteq \mathsf{Inv}(\mathsf{s\text{-}Pol}(\Gamma))$. Moreover, it follows that $\mathsf{s\text{-}Pol}(\Gamma) = \mathsf{s\text{-}Pol}([\Gamma])$.

To establish that $[\Gamma] \supseteq \mathsf{Inv}(\mathsf{s\text{-}Pol}(\Gamma))$, we will show that for any m-ary relation $\varrho \in \mathsf{Inv}(\mathsf{s\text{-}Pol}(\Gamma))$, the relation $\sigma = \varrho \times D^{|D|}$ is a member of $[\Gamma]$. From this it follows that $\varrho \in [\Gamma]$, by applying a sequence of existential quantifications.

To show that $\sigma \in [\Gamma]$, we first define $\sigma' = \bigcap\{\gamma \in [\Gamma] \mid \sigma \subseteq \gamma\}$. Since $[\Gamma]$ contains the total relation $D^{m+|D|}$, and is closed under conjunction, σ' is a member of $[\Gamma]$ and $\sigma \subseteq \sigma'$. In fact, σ' is the minimal relation of arity $m + |D|$ in $[\Gamma]$ with this property (when ordered by inclusion).

Now choose any tuple $\vec{c} = (b_1, \ldots, b_m, d_1, d_2, d_3, \ldots, d_{|D|}) \in \sigma'$. Note that σ' must also contain all tuples of the form $(b_1, \ldots, b_m, d'_1, d'_2, d'_3, \ldots, d'_{|D|})$, for each possible choice of $d'_1, d'_2, d'_3, \ldots, d'_{|D|}$, since otherwise we could obtain a smaller relation σ'' containing σ, by applying a sequence of universal quantifications, followed by a conjunction with the total relation $D^{m+|D|}$. Hence we may choose $\vec{c}$ so that the values of the d_i are all distinct, that is, $\{d_1, d_2, \ldots, d_{|D|}\} = D$.

By Definition 3.2, $[\Gamma]$ is closed under conjunction and existential quantification, and contains the equality relation, $=_D$. It is well-known [30], that such sets satisfy the condition $[\Gamma] = \mathsf{Inv}(\mathsf{Pol}([\Gamma]))$. Furthermore, Proposition 1.1.19 of [30] implies that

$$\sigma' = \{f(\vec{a}_1, \ldots, \vec{a}_n) \mid n \ge 1,\ \vec{a}_1, \ldots, \vec{a}_n \in \sigma,\ f \in \mathsf{Pol}([\Gamma])\}.$$

Therefore, there exist $n \geq 1$, $\vec{a}_1, \ldots, \vec{a}_n \in \sigma$ and an n-ary function $f \in \mathsf{Pol}([\Gamma])$ such that $\vec{c} = f(\vec{a}_1, \ldots, \vec{a}_n)$.

By the choice of $\vec{c}$, the function f must be surjective. Therefore f is in $\mathsf{s\text{-}Pol}([\Gamma])$, and so $f \in \mathsf{s\text{-}Pol}(\Gamma)$, by the observation above. By the choice of ϱ, this implies that ϱ is invariant under f, and so $(b_1, \ldots, b_m) \in \varrho$. It follows that $\sigma' = \sigma$, so $\sigma \in [\Gamma]$, as required.

A.3 Proof of Proposition 4.1

Let $\mathcal{P}$ be an instance of QCSP($\{N\}$), with variables $v_1, v_2, \ldots, v_n$. We construct a corresponding instance $\mathcal{P}'$ of QCSP($\neq_D$) as follows.

First construct a graph, $G_{\mathcal{P}}$, as shown in Figure 1, with 3 nodes for each variable v_i of $\mathcal{P}$ (labelled x_i, y_i and z_i), 3 nodes for each triple of variables constrained by N in $\mathcal{P}$, and one additional node (labelled w). Connect these nodes as indicated in Figure 1, so that each z_i is connected to y_i, each y_i is connected to x_i and w, and each x_i is connected to w. Add edges from each triple of nodes representing a constraint on $v_{i_1}, v_{i_2}, v_{i_3}$ to the corresponding nodes $x_{i_1}, x_{i_2}, x_{i_3}$.

The standard 3-colouring problem for the graph $G_{\mathcal{P}}$ can be represented as a conjunction of disequality constraints, $\neq_D$, with a variable corresponding to each node, and binary disequality constraints for each edge. (If D contains more than 3 values, then we add a clique C containing $|D| - 3$ nodes to the graph $G_{\mathcal{P}}$ and connect each node in the original graph to each node of this clique. This ensures that each node in the original graph $G_{\mathcal{P}}$ must be coloured with one of the 3 colours not used to colour C.)

Now add quantifiers to this conjunction of constraints as follows. First existentially quantify the variable w (and all the variables corresponding to nodes of the clique C, if present). Note that once values have been assigned to these nodes there are just two remaining possible values for each node x_i and y_i.

Next, for each consecutive quantifier of $\mathcal{P}$ (in order) introduce 3 consecutive quantifiers as follows:

- For each existential quantifier in $\mathcal{P}$, $\exists v_i$, introduce $\exists z_i \exists y_i \exists x_i$;
- For each universal quantifier in $\mathcal{P}$, $\forall v_i$, introduce $\forall z_i \exists y_i \exists x_i$.

Finally, add existential quantifiers for all remaining variables (corresponding to the constraints of $\mathcal{P}$). This completes the definition of $\mathcal{P}'$.

It is straightforward to check that there is an assignment of Boolean values to the variables $v_1, v_2, \ldots, v_n$ satisfying all of the constraints of $\mathcal{P}$ if and only if there is an assignment of values from D to the variables of $\mathcal{P}'$ satisfying all the constraints of $\mathcal{P}'$. This is because to satisfy the constraints of $\mathcal{P}'$, the 3 nodes in each triangle in $G_{\mathcal{P}}$ corresponding to a constraint of $\mathcal{P}$ must all be assigned distinct values, which is possible if and only if the corresponding nodes x_{i_1}, x_{i_2} and x_{i_3} connected to them do not all take the same value. Furthermore, the construction of the quantifiers in $\mathcal{P}'$ ensures that the sentence $\mathcal{P}'$ is valid if and only if $\mathcal{P}$ is valid. To see this, note that for any variable v_i of $\mathcal{P}$ which is universally quantified, the universal quantification on the corresponding z_i in $\mathcal{P}'$ forces y_i to take both remaining available values, which mimics the universal quantification on v_i.

Hence, we have established a reduction from QCSP($\{N\}$) to QCSP($\{\neq_D\}$), and it is clear that this reduction can be carried out in logarithmic space. Since QCSP($\{N\}$) is **PSPACE**-complete, by Corollary 2.7, the result follows.

A.4 Proof of Proposition 5.3

Clearly $H \subseteq H^{\perp\perp}$. To prove the equality, let $(g_1, \ldots, g_n) \in G^n \setminus H$. We will show that there are $r_1, \ldots, r_n \in H^{\perp}$ with $\sum_{i=1}^{n} r_i g_i \neq 0$, and so $(g_1, \ldots, g_n) \notin H^{\perp\perp}$.

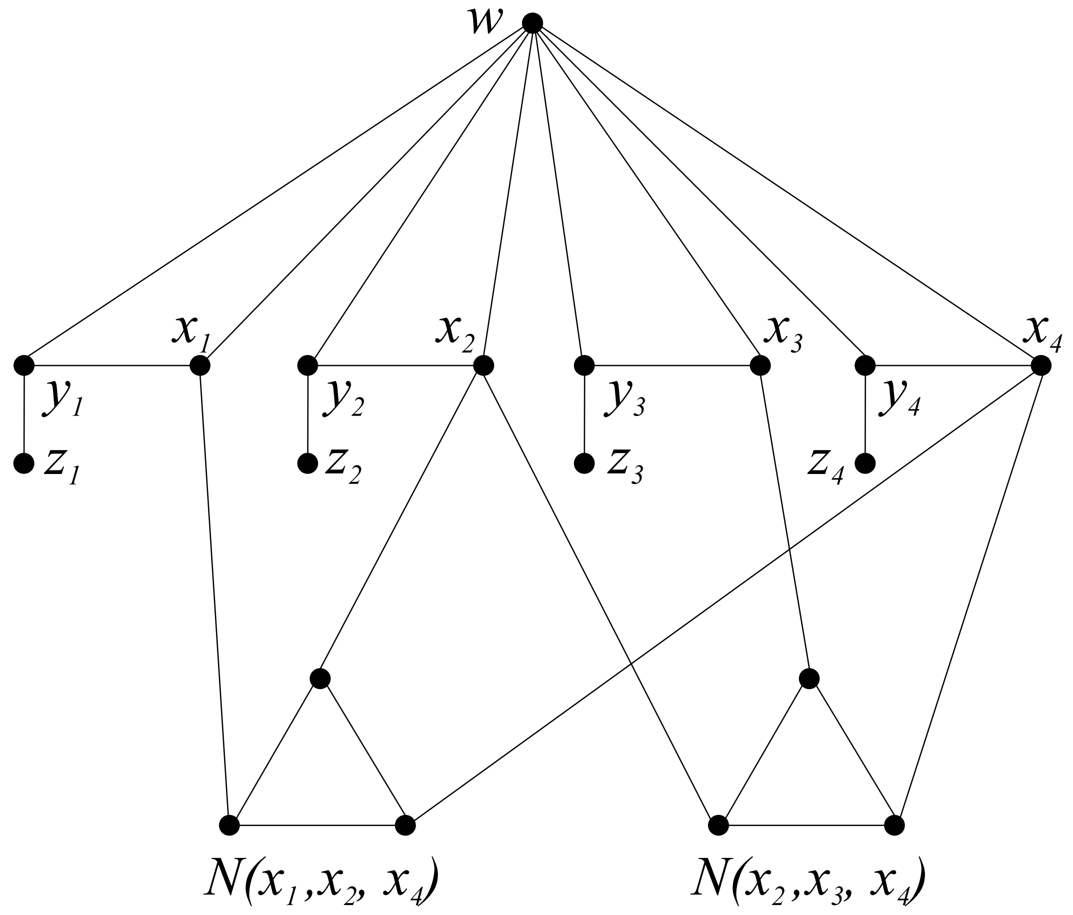

Figure 1: The construction used in the proof of Proposition 4.1 (adapted from Figure 9.8 in [28])

Since H is a subgroup of the finite Abelian group G^n, it follows that G^n also has a subgroup isomorphic to G^n/H (see Theorem 10.57 in [33]). Let $\epsilon : G^n/H \hookrightarrow G^n$ be a corresponding embedding of G^n/H into G^n, and let $\mu : G^n \to G^n/H$ be the canonical homomorphism. Then the function $s := \epsilon \circ \mu : G^n \to G^n$ is a homomorphism with the property

$$s(h_1, \ldots, h_n) = \underline{0} \iff (h_1, \ldots, h_n) \in H \ .$$

Consequently, if we set $(e_1, \ldots, e_n) = s(g_1, \ldots, g_n)$, then we have $(e_1, \ldots, e_n) \neq \underline{0}$, so there exists $k \in \{1, \ldots, n\}$ with $e_k \neq 0$. Let $p_k : G^n \to G$ denote the projection onto the kth component, and let $t = p_k \circ s$. The mapping t is a homomorphism from G^n to G with $t(h_1, \ldots, h_n) = 0$ for all $(h_1, \ldots, h_n) \in H$ and $t(g_1, \ldots, g_n) = e_k \neq 0$.

Now consider the embeddings

$$q_i : G \hookrightarrow G^n, g \mapsto (0, \ldots, 0, \underset{i}{g}, 0, \ldots, 0)$$

and define $r_i : G \to G$ by $r := t \circ q_i$. Then $r_i \in \mathsf{End}\ G$ and we have

$$\sum_{i=1}^{n} r_i h_i = \sum_{i=1}^{n} t(q_i h_i) = t\left(\sum_{i=1}^{n} q_i h_i\right) = t(h_1, \ldots, h_n) = 0$$

for all $(h_1, \ldots, h_n) \in H$, and

$$\sum_{i=1}^{n} r_i g_i = \sum_{i=1}^{n} t(q_i g_i) = t\left(\sum_{i=1}^{n} q_i g_i\right) = t(g_1, \ldots, g_n) = e_k \neq 0$$

This implies that $(g_1, \ldots, g_n) \notin H^{\perp\perp}$, which completes the proof.

A.5 Proof of Corollary 5.4

We first recall some basic facts about the structure of Abelian groups and their endomorphisms. Every finite Abelian group G can be represented as the direct sum $G_1 \oplus \ldots \oplus G_k$ of cyclic subgroups of prime power order: that is, any element $g \in G$ is uniquely representable in the form $g_1 + \ldots + g_k$, with $g_i \in G_i$. This fact we indicate as $g = (g_1, \ldots, g_k)$.

Note that each of the cyclic groups in a decomposition can be viewed as the group of residues modulo a certain integer. We can therefore represent elements of G as tuples of residues with componentwise addition.

Fix a direct decomposition $G = G_1 \oplus \ldots \oplus G_k$, where $G_1, \ldots, G_k$ are cyclic, fix a generator e_i of each G_i, $i = 1, \ldots, k$, and let N be the order of G.

Observation A.1 *For any $g = (g_1, \ldots, g_k) \in G$, $g_i = \alpha_i e_i$, and any $r \in \mathsf{End}\ G$,*

$$rg = rg_1 + \ldots + rg_k = \alpha_1 re_1 + \ldots + \alpha_k re_k.$$

Therefore, r is uniquely determined by the values $re_1, \ldots, re_k$.

Observation A.1 implies that any endomorphism r of G corresponds to a $\mathbb{Z}_N$-matrix (r_{ij}) such that

$$rg = \begin{pmatrix} r_{11} & \cdots & r_{1k} \\ \vdots & & \vdots \\ r_{k1} & \cdots & r_{kk} \end{pmatrix} \begin{pmatrix} g_1 \\ \vdots \\ g_k \end{pmatrix},$$

As is easily seen, the addition and multiplication of endomorphisms correspond to matrix addition and multiplication. Therefore $\mathsf{End}\, G$ is representable as a matrix ring over $\mathbb{Z}_N$.

Take any $\vec{a} \in \varrho$ and set $H = \{\vec{b} - \vec{a} \mid \vec{b} \in \varrho\}$. The set H is a subgroup of G^n; to obtain a generating set for H we employ the following algorithm:

Step 1. Set $M = \emptyset$.
Step 2. Choose an element $\overline{u}$ of H that does not belong to the subgroup generated by M.
Step 3. If such an element does not exist, **then Output** M **and Stop**
Otherwise, set $M := M \cup \{\overline{u}\}$ **and Goto Step 2**.

To fulfil Step 2 of this algorithm we just have to run through H looking for an element $\overline{u}$ for which the equation

$$\overline{u}_1 x_1 + \ldots + \overline{u}_m x_m = \overline{u}$$

is inconsistent, where $M = \{\overline{u}_1, \ldots, \overline{u}_m\}$ and the unknowns take values from $\mathbb{Z}_N$. The consistency of this equation can be checked using standard number-theoretic algorithms (see, e.g., [18]).

As is well known, if G is the direct sum of k cyclic groups of prime power exponent, the size of an (irreducible) generating set of any its subgroups cannot exceed k. This implies that $|M| \leq kn$. Let $M = \{\overline{u}_1, \ldots, \overline{u}_m\}$.

To find a generating set of the $\mathbb{Z}_N$-module $H^\perp$ we need to solve the equation

$$\overline{x}_1 \overline{u}_1 + \ldots + \overline{x}_m \overline{u}_m = 0$$

where $\overline{x}_i = (x_{i1}, \ldots, x_{in}) \in (\mathsf{End}\, G)^n$. This system of equations can be solved by reducing it to a system of linear congruences modulo N, replacing unknown endomorphisms with matrices, and elements of G with k-tuples, as described above. The number of congruences obtained in this way is nk, and the number of unknowns is $k^2 mn$. Hence the linear system of equations obtained is polynomial in size, and can be solved by standard number-theoretic algorithms [18] in polynomial time.

Now, let $\vec{r}_1, \ldots, \vec{r}_m$ be a generating set of the submodule $H^\perp$, where each $\vec{r}_i = (r_{i1}, \ldots, r_{in})$. Any tuple $(g_1, \ldots, g_n) \in H$ satisfies the system

$$\begin{array}{rcl} r_{11}x_1 + \ldots + r_{1n}x_n & = & 0 \\ & \vdots & \\ r_{m1}x_1 + \ldots + r_{mn}x_n & = & 0. \end{array} \tag{1}$$

Since $\vec{r}_1, \ldots, \vec{r}_m$ generate the $\mathsf{End}\, G$-module $H^\perp$, a tuple $(g_1, \ldots, g_n) \in G^n$ satisfies the system of equations (1) if and only if it satisfies any equation of the form $q_1 x_1 + \ldots + q_n x_n = 0$, $(q_1, \ldots, q_n) \in H^\perp$, and hence, by Proposition 5.3, belongs to H. Finally, $\overline{b} \in \varrho$ if and only if $\overline{b} - \overline{a} \in H$, and, therefore, if and only if $\overline{b}$ satisfies the following system of equations

$$\begin{array}{rcl} r_{11}x_1 + \ldots + r_{1n}x_n & = & c_1 \\ & \vdots & \\ r_{m1}x_1 + \ldots + r_{mn}x_n & = & c_m \end{array}$$

where $c_i = r_{i1}a_1 + \ldots + r_{in}a_n$.

Input: A sentence S of the form $\mathcal{Q}_1 x_1 \ldots \mathcal{Q}_l x_l \ (\vec{r}_1 x_1 + \ldots + \vec{r}_l x_l \in H + \vec{b})$.
Output: Whether S is valid.

Step 1. **For each** $i = l, \ldots, 1$ **do**
  If x_i is existentially quantified **then**
    Remove the summand $\vec{r}_i x_i$ from the left side, and
    Replace the subgroup H in the right side with the subgroup generated by H and $\vec{r}_i G$.
  If x_i is universally quantified **then**
    If $\vec{r}_i G \subseteq H$, **then remove** the summand $\vec{r}_i x_i$ from the left side.
    Otherwise, Output 'No' and **Stop**.
Step 2. **If** $0 \in H + \vec{b}$ **then Output** 'Yes', **otherwise Output** 'No'.

Figure 2: Algorithm for deciding QCSP(Γ) for affine Γ

A.6 Proof of Proposition 5.5

For the sake of conciseness, we will write linear systems of equations in vector form, as follows

$$\vec{r}_1 x_1 + \ldots + \vec{r}_l x_l = \vec{b}$$

where $\vec{r}_1, \ldots, \vec{r}_l$ are elements of $(\mathsf{End}\ G)^m$, and m is the number of equations.

We now consider how such linear systems of equations can be solved. When solving equations in number theory it is often useful to transform a congruence of the form $(ac)x \equiv ad \pmod{ab}$ to the congruence $cx \equiv d \pmod{b}$. In our more general setting, such a transformation corresponds to considering a linear system modulo a subgroup $H \leq G^n$. We therefore consider more general linear systems of equations of the following form

$$\vec{r}_1 x_1 + \ldots + \vec{r}_l x_l \in H + \vec{b}$$

where the right-hand side now specifies a coset of H in G^n, rather than a fixed element.

In this more general framework, a sentence of the form given in the Proposition is equivalent to a sentence of the form

$$\mathcal{Q}_1 x_1 \ldots \mathcal{Q}_l x_l \quad (\vec{r}_1 x_1 + \ldots + \vec{r}_l x_l \in \{0\} + \vec{b}).$$

To decide whether this sentence holds, we apply the algorithm shown in Figure 2. (In the description of the algorithm, we use $\vec{r}G$, where $\vec{r} = (r_1, \ldots, r_m) \in (\mathsf{End}\ G)^m$, to denote the set $\{(r_1 g, \ldots, r_m g) \mid g \in G\}$.)

To prove the soundness of the algorithm we note that at each iteration, when the algorithm deals with the variable x_i, the coset $H + \vec{b}$ is the set of all values $\vec{a}$ such that, for any assignment to the universally quantified variables, there is an assignment to the existentially quantified variables with $\vec{r}_{i+1} x_{i+1} + \ldots + \vec{r}_l x_l = \vec{a}$. Therefore, if x_i is universally quantified, and $\vec{r}_i G \not\subseteq H$ then, for any assignment to $x_1, \ldots, x_{i-1}$ there exists $g \in G$ such that $\vec{r}_1 x_1 + \ldots + \vec{r}_{i-1} x_{i-1} + \vec{r}_i g \notin H + \vec{b}$ and the input sentence is false. Finally, denoting the coset obtained after finishing Step 1 by $H + \vec{b}$, the sentence holds if and only if, for any assignment of values to the universally quantified variables, there is an assignment to the existentially quantified variables that makes the left side equal $\vec{b}$, or equivalently, $0 \in H + \vec{b}$.

As we have shown above, all calculations with group elements and endomorphisms can be performed by making use of standard number-theoretic algorithms, using the fact that G is a direct product of subgroups, each of which is a group of integers modulo a prime power. Hence this algorithm can be completed in polynomial time.

A.7 Proof of Proposition 5.10

To establish this result we show that when the implication graph is both invertible and sufficient, then the associated problem instance can be decided simply by checking for the following simple property of this graph, which we will call *consistency*.

Definition A.2 *Let $\mathcal{P}$ be a QCSP instance, and let* $\mathrm{I}(\mathcal{P})$ *be the implication graph of* $\mathcal{P}$.

A node (v_i, a) *of* $\mathrm{I}(\mathcal{P})$ *will be called* consistent *if there is no path from* (v_i, a) *to* (v_i, b) *for any* $b \neq a$.

The graph $\mathrm{I}(\mathcal{P})$ *will be called* consistent *if the following conditions hold for all choices of variables* v_i *and* v_j *and values* a *and* b*:*

1. *If* v_i *is universally quantified, then* $\mathrm{I}(\mathcal{P})$ *contains a consistent node* (v_i, a) *for every possible value* a.
2. *If* v_i *is existentially quantified, then* $\mathrm{I}(\mathcal{P})$ *contains a consistent node* (v_i, a) *for some possible value* a.
3. *If* v_i *is existentially quantified, and* v_j *is universally quantified, and* $j > i$*, then there is no cycle containing both* (v_i, a) *and* (v_j, b).
4. *If* v_i *and* v_j *are both universally quantified, and* $v_i \neq v_j$*, then there is no path from* (v_i, a) *to* (v_j, b).

Note that checking whether a given ordered pair of vertices is an arc in $\mathrm{I}(\mathcal{P})$ can be done in logarithmic space. By Definition A.2, $\mathrm{I}(\mathcal{P})$ is *not* consistent if and only if certain paths are present in $\mathrm{I}(\mathcal{P})$. Checking for the existence of paths in a graph is known to be in **NL**, and **NL** is closed under complementation, so we conclude that the problem of checking whether $\mathrm{I}(\mathcal{P})$ is consistent is also in **NL**.

Hence, the following lemma completes the proof.

Lemma A.3 *Let* $\mathcal{P}$ *be a QCSP instance whose implication graph* $\mathrm{I}(\mathcal{P})$ *is both invertible and sufficient. The instance* $\mathcal{P}$ *is valid if and only if* $\mathrm{I}(\mathcal{P})$ *is consistent.*

Proof: Let $\mathcal{P} = (V, D, F)$ be an instance of QCSP such that $\mathrm{I}(\mathcal{P})$ is both invertible and sufficient.

Assume first that $\mathrm{I}(\mathcal{P})$ is not consistent. There are four cases to consider, corresponding to the four conditions in Defintion A.2.

1. If there is some universally quantified variable v_i such that (v_i, a) is inconsistent for some a, then there are no models with $v_i = a$, and hence $\mathcal{P}$ is not valid.
2. If there is some existentially quantified variable v_i such that (v_i, a) is inconsistent for every a, then there are no models with any value for v_i, and hence $\mathcal{P}$ is not valid.
3. If there is some existentially quantified variable v_i, and some universally quantified variable v_j, such that $j > i$, and there is a cycle containing both (v_i, a) and (v_j, b), then, by invertibility, for each $a' \neq a$ there is a path from (v_i, a') to (v_j, b'), for some $b' \neq b$. Hence every possible assignment to v_i forces a particular assignment to v_j, which implies that $\mathcal{P}$ is not valid.
4. If there are two distinct universally quantified variables v_i and v_j, such that there is a path from (v_i, a) to (v_j, b) for some a and b, then the assignment of a to v_i forces the assignment of b to v_j, and hence $\mathcal{P}$ is not valid.

Hence if $\mathrm{I}(\mathcal{P})$ is not consistent then $\mathcal{P}$ is not valid.

To establish the converse, assume that $\mathrm{I}(\mathcal{P})$ is consistent. We will show that the algorithm described below can be used to prove that $\mathcal{P}$ is valid.

We first define some terminology. If there is a path from node (v_i, a) to a distinct node (v_j, b) then we shall say that (v_i, a) is *upstream* of (v_j, b) and (v_j, b) is *downstream* of (v_i, a). If there is a path in both directions then we shall say that (v_i, a) and (v_j, b) are *equivalent*. If v_i is universally quantified, then v_i will be called a *universal variable*, and any node (v_i, a) will be called a *universal node*. Similarly, if v_i is existentially quantified, then v_i will be called an *existential variable*, and any node (v_i, a) will be called an *existential node*.

Note that if some existential node (v_i, a) is equivalent to a universal node (v_j, b), then, by invertibility, every node (v_i, a') is upstream of a node (v_j, b') and every node (v_j, b') is upstream of a node (v_i, a'). Furthermore, each node (v_j, b') must be equivalent to exactly one node (v_i, a'), otherwise there will be a path between two distinct universal nodes and $\mathrm{I}(\mathcal{P})$ will be inconsistent. Hence $\mathrm{I}(\mathcal{P})$ contains a node (v_i, a') for each possible value a', and this node is equivalent to (v_j, b') for some b'. When this condition applies we say that v_i is equivalent to v_j.

Now consider the following algorithm:

```
(1)  For each variable v_i
(2)      If v_i is universal or v_i is equivalent to some universal variable, then
(3)         For each possible value a
(4)             Mark (v_i, a) contingent
(5)      Else
(6)         Choose a value a' such that (v_i, a') is consistent and
              not upstream of any universal node or any node marked false
(7)             Mark (v_i, a') true
(8)             For all a'' ≠ a', mark (v_i, a'') false
```

Note that any node (v_i, a'') which is marked **false** by this algorithm cannot be downstream of a node previously marked **true** or a universal node, since if it were, then by invertibility the corresponding node (v_i, a') marked **true** would be upstream of a node previously marked **false** or a universal node, which contradicts the choice of a'.

To show that this algorithm marks each node exactly once it suffices to show that it is always possible to choose a suitable value for a' at line (6).

To show this, assume for contradiction that v_i is the first existential variable selected by the algorithm which is not equivalent to a universal variable and where every consistent node (v_i, a) is upstream of a universal node or a node already marked **false**.

By consistency, since v_i is existential, there must be at least one node (v_i, a) which is consistent. There are two cases to consider.

If (v_i, a) is upstream of a universal node (v_j, b), then, by invertibility, there is a node (v_i, a') with $a' \neq a$ which is downstream of a universal node (v_j, b'). Furthermore, (v_i, a') must be consistent, since if there is a path from (v_i, a') to (v_i, a''), for some $a'' \neq a'$, then by invertibility (v_i, a'') is upstream of a different universal node and hence there is a path from one universal node to another and $\mathrm{I}(\mathcal{P})$ is not consistent. For the same reason, (v_i, a') cannot be upstream of any universal node except (v_j, b'), and we have already excluded the possibility that (v_i, a') is equivalent to (v_j, b'), so we have that (v_i, a') is not upstream of any universal node. Hence, by our assumption about v_i, (v_i, a') must be upstream of a node which is already marked **false**. But this implies that there is a node marked **false** which is downstream of a universal node, which contradicts our earlier observation about nodes marked **false**.

Alternatively, if (v_i, a) is upstream of a node (v_j, b) already marked **false**, then, by invertibility, there is a node (v_i, a') with $a' \neq a$ which is downstream of a node (v_j, b') which is already marked **true**. But, by our assumption about v_i, this implies that (v_j, b') is upstream of a universal node or a node already marked **false**, which contradicts the choice of b'.

Since both cases lead to a contradiction, we conclude that it is always possible to make the choice required in line (6), and hence the algorithm above marks every node.

Once all the nodes are marked, we can construct a corresponding assignment in which universal variables are assigned arbitrarily, then equivalent existential variables (which must have higher subscripts, by consistency) are given the unique values forced by these assignments, and finally the value a is assigned to variable v_i whenever the node (v_i, a) is marked **true**. The construction of the marking ensures that there is no path from a node marked **true** or **contingent** to a node marked **false**, or to a non-equivalent node marked **contingent**, so, by the sufficiency of $\mathrm{I}(\mathcal{P})$, any assignment constructed in this way will be a model of $\mathcal{P}$. Hence $\mathcal{P}$ is valid. ∎

A.8 Proof of Proposition 6.1

For any $\overline{a} = (a_1, \ldots, a_n) \in D^n$, and any subsequence $i_1, \ldots, i_k$ of the sequence $1, \ldots, n$, we define $\mathrm{pr}_{i_1,\ldots,i_k}\overline{a}$ to be the k-tuple $(a_{i_1}, \ldots, a_{i_k})$. Moreover, for any n-ary relation ϱ, we define $\mathrm{pr}_{i_1,\ldots,i_k}\varrho$ to be the k-ary relation

$$\mathrm{pr}_{i_1,\ldots,i_k}\varrho = \{\mathrm{pr}_{i_1,\ldots,i_k}\overline{a} \mid \overline{a} = (a_1, \ldots, a_n) \in \varrho\}.$$

For $I = \{i_1, \ldots, i_k\}$, we will sometimes write $\mathrm{pr}_I\varrho$ instead of $\mathrm{pr}_{i_1,\ldots,i_k}\varrho$. Note that $\mathsf{Pol}(\{\mathrm{pr}_I\varrho\}) \supseteq \mathsf{Pol}(\{\varrho\})$.

We first clarify the structure of relations over a set D which are invariant under the near-projection operation $l_{|D|}$.

Let $\underline{n}$ denote the set $\{1, \ldots, n\}$. Suppose $I_1, \ldots, I_k$ is a partition of $\underline{n}$ and let $\varrho_j = \mathrm{pr}_{I_j}\varrho$ for $j = 1, \ldots, k$. Then we write $\varrho = \varrho_1 \times \ldots \times \varrho_k$ if ϱ can be represented as $\varrho = \{\overline{a} \mid \mathrm{pr}_{I_j}\overline{a} \in \varrho_j \text{ for every } j = 1, \ldots, k\}$.

Lemma A.4 *Let $\varrho \in R_D^{(n)}$, where $|D| \geq 3$.*

If $\varrho \in \mathsf{Inv}(\{l_{|D|}\})$ and $\mathrm{pr}_i\varrho = D$ for every $i \in \underline{n}$, then ϱ is of the form

$$\varrho = \varrho_1 \times \ldots \times \varrho_k$$

where each $\varrho_j = \{(a, \pi_{2j}(a), \ldots, \pi_{m_j j}(a)) \mid a \in D\}$, for some permutations $\pi_{2j}, \ldots, \pi_{m_j j}$ of D.

Proof: We prove the lemma by induction on n. When $n = 1$ the result holds trivially, so consider the case when $n = 2$. Assume that ϱ is not a graph of a permutation. Then there exist $b_1, b_2, b \in D$ such that $b_1 \neq b_2$ and $(b_1, b), (b_2, b) \in \varrho$. Since $\mathrm{pr}_1\varrho = D$, it is possible to choose $\vec{a}_1, \vec{a}_2, \ldots, \vec{a}_{|D|} \in \varrho$ so that $\vec{a}_1(1), \ldots, \vec{a}_{|D|}(1)$ are all different, $\vec{a}_1(1)$ is an arbitrary element from $D \setminus \{b_1, b_2\}$, $\vec{a}_2 = (b_1, b)$, and $\vec{a}_{|D|} = (b_2, b)$. Since ϱ is invariant under $l_{|D|}$, we have $l_{|D|}(\vec{a}_1, \ldots, \vec{a}_k) = (\vec{a}_1(1), b) \in \varrho$, and hence $(x, b) \in \varrho$ for all $x \in D$.

It follows that, for any $(x, y) \in D^2$ we can choose $\vec{c}_1, \vec{c}_2, \ldots, \vec{c}_{|D|} \in \varrho$ such that $\vec{c}_1(1), \ldots, \vec{c}_{|D|}(1)$ are all different, $\vec{c}_1(1) = x$, $\vec{c}_{|D|}(2) = y$, and $\vec{c}_1(2) = \ldots = \vec{c}_{|D|-1}(2) = b$. Since ϱ is invariant under $l_{|D|}$, we have $l_{|D|}(\vec{c}_1, \ldots, \vec{c}_{|D|}) = (x, y) \in \varrho$, and hence $\varrho = D^2$.

We now prove the induction step. By the argument above, for any pair $i, j \in \underline{n}$ the projection $\mathrm{pr}_{i,j}\varrho$ is either D^2, or the graph of a permutation. Assume that there exist i, j such that $\mathrm{pr}_{i,j}\varrho$ is the graph of a permutation π. By the inductive hypothesis $\mathrm{pr}_{\underline{n}\setminus\{j\}}\varrho$ can be represented in the form

$$\mathrm{pr}_{\underline{n}\setminus\{j\}}\varrho = \varrho_1 \times \ldots \times \varrho_k,$$

and the i-th coordinate position occurs in one of $\varrho_1, \ldots, \varrho_k$. Suppose, for simplicity, that i is the last coordinate position in ϱ_1, that is,

$$\begin{aligned}\varrho_1 \;=\; &\{(a_{i_1}, \ldots, a_{i_{m_1-1}}, a_i) \mid a_{i_1} \in D,\ a_{i_s} = \pi_{s1}(a_{i_1}) \\ &\quad \text{for } s \in \{2, \ldots, m_1 - 1\},\ a_i = \pi_i(a_{i_1})\}.\end{aligned}$$

Then, letting

$$\begin{aligned}\varrho_1' &= \{(a_{i_1},\ldots,a_{i_{m_1-1}},a_i,a_j) \mid a_{i_1}\in D,\ a_{i_s}=\pi_{s1}(a_{i_1})\\ &\qquad \text{for } s\in\{2,\ldots,m_1-1\},\ a_i=\pi_i(a_{i_1}), a_j=\pi\pi_i(a_{i_1})\}\end{aligned}$$

we have $\varrho = \varrho_1' \times \varrho_2 \times \ldots \times \varrho_k$, as required.

It remains to prove that if $\mathrm{pr}_{i,j}\varrho = D^2$ for every $i,j \in \underline{n}$, then $\varrho = D^n$.

For any $a \in D$, define

$$\varrho_a = \{(a_1,\ldots,a_{n-1}) \mid (a_1,\ldots,a_{n-1},a) \in \varrho\}.$$

Since the operation $l_{|D|}$ is idempotent, ϱ_a also belongs to $\mathsf{Inv}(\{l_{|D|}\})$.

Consider first the case $n = 3$.

Suppose that, for some $a \in D$, the relation ϱ_a is not the graph of a permutation. Then $\varrho_a = D^2$ by the argument above. Take any $c \in D$ such that $c \neq a$. Then there exists a tuple $\vec{c} = (c_1, c_2, c) \in \varrho$. For $1 \le i \le |D|-1$, choose $\vec{a}_i = (x_i, y_i, a) \in \varrho$, such that $\{x_1,\ldots,x_{|D|-1},c_1\} = \{y_1,\ldots,y_{|D|-1},c_2\} = D$. Then $l_{|D|}(\vec{a}_1,\ldots,\vec{a}_{|D|-1},\vec{c}) = (x_1,y_1,c) \in \varrho$. Since we can change y_1 whilst keeping the same x_1, we conclude that ϱ_c is not the graph of a permutation. Thus $\varrho_c = D^2$ for all $c \in D$, which implies that $\varrho = D^3$.

Now consider the remaining case, where ϱ_a is the graph of a permutation for each $a \in D$. In this case $|\varrho_a| = |D|$ for each a, and since $|\mathrm{pr}_{1,2}\varrho| = |\bigcup_{a\in D}\varrho_a| = |D^2|$, we have $\varrho_a \cap \varrho_b = \emptyset$ for all $a,b \in D$ such that $a \neq b$. Assume that $D = \{d_1, d_2, \ldots, d_{|D|}\}$. For $1 \le i \le |D|-1$, choose $\vec{a}_i = (a_i, d_1, d_i) \in \varrho$, and choose $\vec{a}_{|D|} = (a_1, b, d_{|D|}) \in \varrho$. Note that $a_1,\ldots,a_{|D|-1}$ are all different, and $b \neq d_1$, because $\varrho_{d_i} \cap \varrho_{d_j} = \emptyset$ if $i \neq j$. Now $l_{|D|}(\vec{a}_1,\ldots,\vec{a}_{|D|}) = (a_1,b,d_1) \in \varrho$, so $(a_1,b) \in \varrho_{d_1} \cap \varrho_{d_{|D|}}$, a contradiction.

If $n > 3$ then, by the inductive hypothesis, we have $\mathrm{pr}_{i,j,n}\varrho = D^3$ and, consequently, $\mathrm{pr}_{i,j}\varrho_a = D^2$ holds for all $1 \le i,j \le n-1$. Applying the inductive hypothesis to ϱ_a, we obtain $\varrho_a = D^{n-1}$ for each $a \in D$, which implies that $\varrho = D^n$. ∎

Lemma A.5 *Let $\varrho \in R_D^{(n)}$, where $|D| \ge 3$.*

If $\varrho \in \mathsf{Inv}(\{l_{|D|}\})$ and $I = \{i \in \underline{n} \mid |\mathrm{pr}_i\varrho| < |D|\}$, then $\varrho = \mathrm{pr}_I\varrho \times \mathrm{pr}_{\underline{n}\setminus I}\varrho$.

Proof: By Lemma A.4, $\mathrm{pr}_{\underline{n}\setminus I}\varrho = \varrho_1 \times \ldots \times \varrho_k$ where $\varrho_j = \{(a, \pi_{2j}(a), \ldots, \pi_{m_j j}(a)) \mid a \in D\}$ and $\pi_{2j},\ldots,\pi_{m_j j}$ are permutations of D. Denote the set of coordinate indices of ϱ_j by J_j, and let J_0 be a system of representatives of $J_1,\ldots,J_k$. Then $\mathrm{pr}_{J_0}\varrho = D^{|J_0|}$.

Take an arbitrary $\vec{b} \in \mathrm{pr}_I\varrho$ and $\vec{c} \in \mathrm{pr}_{J_0}\varrho$. There exists $\vec{a}_{|D|} \in \mathrm{pr}_{I\cup J_0}\varrho$ such that $\mathrm{pr}_I\vec{a}_k = \vec{b}$. For $1 \le i \le |D|-1$, choose $\vec{a}_i \in \mathrm{pr}_{I\cup J_0}\varrho$ such that for each $j \in J_0$, $\{\vec{a}_i(j) \mid 1 \le i \le |D|\} = D$. (This is possible because $\mathrm{pr}_{J_0}\varrho = D^{|J_0|}$.)

Now let $\vec{d} = l_{|D|}(\vec{a}_1,\ldots,\vec{a}_{|D|}) \in \mathrm{pr}_{I\cup J_0}\varrho$. It is easy to check that $\mathrm{pr}_I\vec{d} = \vec{b}$ and $\mathrm{pr}_{\underline{n}\setminus I}\vec{d} = \vec{c}$. Hence, $\mathrm{pr}_{I\cup J_0}\varrho = \mathrm{pr}_I\varrho \times \mathrm{pr}_{J_0}\varrho$. Finally, by the choice of J_0, any element from $\mathrm{pr}_{I\cup J_0}\varrho$ has a unique extension to an element of ϱ, and the result follows. ∎

Now we are in a position to describe a polynomial-time reduction from QCSP(Γ) to CSP($\mathsf{Inv}(\mathsf{Pol}(\Gamma))$).

Let $\mathcal{P} = \mathcal{Q}_1x_1\ldots\mathcal{Q}_lx_l\ \Phi$ be an instance of QCSP(Γ) where $\Phi = \varrho_1 \wedge \ldots \wedge \varrho_q$. Suppose that some variable x occurs in ϱ_s at position i and in ϱ_t at position j. If φ is a model of Φ, then, clearly $\varphi(x)$ must belong to both $\mathrm{pr}_i\varrho_s$ and $\mathrm{pr}_j\varrho_t$. We can therefore remove from both ϱ_s and ϱ_t all the tuples whose coordinate in the position of x is not in $\mathrm{pr}_i\varrho_1 \cap \mathrm{pr}_j\varrho_2$, and this procedure does not change the set of models of Φ. Suppose that ϱ_s' and ϱ_t' are the relations obtained from ϱ_s and ϱ_t by performing this procedure. We can substitute ϱ_s' and ϱ_t' for ϱ_s and ϱ_t in $\mathcal{P}$, and the obtained instance will be equivalent to $\mathcal{P}$. It is easy to see that ϱ_s' and ϱ_t' can be obtained from ϱ_1 and ϱ_2 by using conjunction and existential quantification, and hence, by Theorem 2.3.1 of [30], $\varrho_s', \varrho_t' \in \mathsf{Inv}(\mathsf{Pol}(\Gamma))$.

Let $P' = \mathcal{Q}_1 x_1 \ldots \mathcal{Q}_l x_l \; \Phi'$ be the instance obtained from $\mathcal{P}$ by performing the above procedure for all variables and pairs of atomic formulas in $\mathcal{P}$ until the process is stable[1]. Clearly $\mathcal{P}'$ can be obtained in polynomial time.

If one of the predicates in Φ' is always false, or if some variable that cannot take all values is universally quantified in $\mathcal{P}'$, then clearly $\mathcal{P}'$ is not valid, and so neither is $\mathcal{P}$. Otherwise, by Lemmas A.4 and A.5, Φ' can be represented as $\Phi_1 \wedge \Phi_2$ so that Φ_1 and Φ_2 have no variable in common, Φ_1 is a conjunction of graphs of permutations, and no variable in Φ_2 can take all possible values. Hence $\mathcal{P}'$ can be represented as a conjunction of two sentences: one (corresponding to Φ_1) is an instance of $\mathrm{QCSP}(\Delta)$, and the other is an instance of $\mathrm{CSP}(\mathsf{Inv}(\mathsf{Pol}(\Gamma)))$. It is easy to check that $\Delta \subseteq \mathsf{Inv}(d)$. Hence, by Theorem 5.7, we can solve any instance of $\mathrm{QCSP}(\Delta)$ in polynomial time, and we have reduced $\mathrm{QCSP}(\Gamma)$ to $\mathrm{CSP}(\mathsf{Inv}(\mathsf{Pol}(\Gamma)))$.

[1] This is a standard pre-processing procedure in constraint satisfaction problems, which is referred to as "establishing arc-consistency"