The complexity of counting quantifiers on equality languages[☆]

Barnaby Martin[a], András Pongrácz[b,*], Michał Wrona[c,**]

[a] *School of Engineering & Computing Sciences, Durham University, Durham, UK.*
[b] *Debreceni Egyetem TTK, Algebra és Számelmélet Tanszék, 4010 Debrecen, Pf. 18, Hungary.*
[c] *Theoretical Computer Science Department, Faculty of Mathematics and Computer Science, Jagiellonian University, Kraków, Poland.*

Abstract

An equality language is a relational structure with infinite domain whose relations are first-order definable in equality. We classify the extensions of the quantified constraint satisfaction problem over equality languages in which the native existential and universal quantifiers are augmented by some subset of counting quantifiers. In doing this, we find ourselves in various worlds in which dichotomies or trichotomies subsist.

Keywords: Quantified Constraints, Counting Quantifiers, Constraint Satisfaction, Equality Language, Computational Complexity

1. Introduction

The *constraint satisfaction problem* CSP, much studied in artificial intelligence, is known to admit several equivalent formulations, two of the best known of which are the query evaluation of primitive positive (pp) sentences – those involving only existential quantification and conjunction – and the homomorphism problem (see, e.g., [1]). The CSP is NP-complete in general, and a great deal of effort has been expended in classifying the complexity of CSP(Γ) across fixed, finite *constraint languages* Γ. Notably it is conjectured [2, 3] that for all such finite Γ, the problem CSP(Γ) is in P or NP-complete.

[☆] An extended abstract of this paper appeared at Computability in Europe (CiE) 2016.
[*] Supported by the Hungarian Scientific Research Fund (OTKA) grant no. K109185.
[**] Partially supported by NCN grant number 2014/14/A/ST6/00138.

While this has not been settled in general, a number of partial results are known – e.g. over structures of size at most three [4, 5] and over smooth digraphs [6, 7].

A popular generalisation of the CSP involves considering the query evaluation problem for *positive Horn* logic – involving only the two quantifiers, $\exists$ and $\forall$, together with conjunction. The resulting *quantified constraint satisfaction problems* QCSP(Γ) allow for a broader class, used in artificial intelligence to capture non-monotonic reasoning [8], whose complexities rise to Pspace-completeness.

Once upon a time, Bodirsky and Kára gave a systematic classification for CSP(Γ), where Γ consists of relations first-order (fo) definable in equality, over some countably infinite domain [9]. These so-called *equality languages* Γ display dichotomy between those for which CSP(Γ) is in P and those for which it is NP-complete. As explained in [10], equality languages form a base case in the pursuit of grander classifications across first-order definitions over more complicated structures. Pursuing this line of investigation, Bodirsky and Chen gave a trichotomy for QCSP(Γ), where Γ is an equality language – each problem being either in P, NP-complete or co-NP-hard [11]. In the conference version of that paper, the trichotomy was claimed to be across P, NP-complete or Pspace-complete [12], but the proof in the tricky case of $x = y \to y = z$ was flawed, and so in the journal version this became the weaker co-NP-hard (and in Pspace). The trichotomy is thus imperfect, as most of the co-NP-hard cases are known to be Pspace-complete. Indeed, $x = y \to y = z$ would be the only open case, if it were Pspace-complete [13].

Working Hypothesis. QCSP($x = y \to y = z$) is Pspace-complete.

Thus the assumption of the working hypothesis would restore the trichotomy to the P, NP-complete or Pspace-complete as stated in [12].

In this paper, we consider the generalisation of the QCSP with counting quantifiers, as pioneered in the recent paper [14]. In [14], the domains of Γ were of finite size n, so the extant quantifiers $\exists^{\geq 1} = \exists$ and $\exists^{\geq n} = \forall$ were augmented with quantifiers of the form $\exists^{\geq j}$, which allow one to assert the existence of at least j elements such that the ensuing property holds. In the world of infinite domains, it makes sense to permit not only quantification above the finite with $\exists^{\geq j}$, but also quantification *below the co-finite* with $\forall^{>j}$, whose intended meaning is that the property holds for all but (at most) j elements of the domain. Thus, $\forall = \forall^{>0}$. Counting quantifiers of the form $\exists^{\geq j}$ have been extensively studied in finite model theory (see [15, 16]),

where the focus is on supplementing the descriptive power of various logics. Quantifiers of the form $\forall^{>j}$ appear rare in computer science but these quantifiers together with $\exists^{\geq j}$ are termed *hemilogical* when they appear in [17]. Of broader interest is the *majority quantifier* $\exists^{\geq n/2}$ (on a structure of domain size n), which sits broadly midway between $\exists$ and $\forall$. Majority quantifiers are studied across diverse fields of logic and have various practical applications, e.g. in cognitive appraisal and voting theory [18, 19]. They have also been studied in computational complexity since at least [20] (see also [15]).

We study extensions of QCSP(Γ) in which the input sentence to be evaluated on Γ remains positive conjunctive in its quantifier-free part, but is quantified by various counting quantifiers. For $X \subseteq \{\exists^{\geq 1}, \exists^{\geq 2}, \ldots, \forall^{>0}, \forall^{>1}, \ldots\}$, $X \supseteq \{\exists^{\geq 1}, \forall^{>0}\}$, the X-CSP(Γ) takes as input a sentence given by a conjunction of atoms quantified by quantifiers appearing in X. It then asks whether this sentence is true on Γ.

Equality languages admit quantifier elimination of $\forall$ and $\exists$, that is any relation first-order definable in equality is already quantifier-free definable, say as a CNF. An equality language Γ is

- *trivial* if all its relations may be given as a conjunction of equalities,
- *specially negative* if the class of relations over Γ, closed under definability in the positive conjunctive logic with quantifiers among $\{\exists, \forall, \forall^{>1}\}$, does not contain the formula $x \neq y \vee y \neq z$,
- *negative* if all its relations may be given as a conjunction of equalities and disjunctions of disequalities, and
- *positive* if all its relations may be given as a conjunction of disjunctions of equalities.

Similarly, we might use these adjectives on the relations within the equality language. We observe the containments of trivial languages within specially negative languages within negative languages. Further, it is proved in [11] (Proposition 7.3) that the positive languages that are not trivial are precisely the positive languages that are not negative.

Our main results are a complete panoply of classifications for $X \supseteq \{\exists^{\geq 1}, \forall^{>0}\}$, that is the augmentation of $\exists$ and $\forall$ with the more exotic counting quantifiers. It will be seen that the quantifiers $\exists^{\geq 2}, \exists^{\geq 3}, \ldots$ more or less behave as one another and similarly with $\forall^{>2}, \forall^{>3}, \ldots$. However, $\forall^{>1}$ is special and thus our

task of classifications for X amounts to choosing subsets of $\{\exists^{\geq 2}, \forall^{>1}, \forall^{>2}\}$ with which to augment $\{\exists^{\geq 1}, \forall^{>0}\}$. A priori there are then eight possibilities, but twice we will see $\forall^{>1}$ being "subsumed" by $\forall^{>2}$. Thus we will give *six distinct classification theorems*: three dichotomies and three trichotomies (one of which is that of [11]). In Figure 1, these classification theorems are linked to their canonical subsets of $\{\exists^{\geq 2}, \forall^{>1}, \forall^{>2}\}$.

Theorem 1 ([11]). *If $X = \{\exists^{\geq 1}, \forall^{>0}\}$, then X-CSP(Γ) displays **trichotomy** on the class of equality languages Γ:*

- *if all relations of Γ are **negative** then X-CSP(Γ) is in L.*
- *if all relations of Γ are **positive** but some relation is not trivial, then X-CSP(Γ) is NP-complete.*
- ***otherwise** X-CSP(Γ) is co-NP-hard.*

Theorem 2. *If $\{\exists^{\geq 1}, \forall^{>0}\} \subseteq X \subseteq \{\exists^{\geq 1}, \forall^{>0}, \forall^{>1}, \forall^{>2}, \ldots\}$ and contains some $\forall^{>j}$ for $j \geq 2$, then X-CSP(Γ) displays **trichotomy** on the class of equality languages Γ:*

- *if all relations of Γ are **trivial**, then X-CSP(Γ) is in L.*
- *if all relations of Γ are **positive**, but some relation is not trivial, then X-CSP(Γ) is NP-complete.*
- ***otherwise** X-CSP(Γ) is Pspace-complete.*

Theorem 3. *If $X = \{\exists^{\geq 1}, \forall^{>0}, \forall^{>1}\}$, then X-CSP(Γ) displays **trichotomy** on the class of equality languages Γ:*

- *if Γ is **specially negative**, then X-CSP(Γ) is in P.*
- *if all relations of Γ are **positive**, but some relation is not trivial, then X-CSP(Γ) is NP-complete.*
- ***otherwise** X-CSP(Γ) is Pspace-complete.*

Theorem 4. *If $\{\exists^{\geq 1}, \forall^{>0}\} \subseteq X \subseteq \{\forall^{>0}, \exists^{\geq 1}, \exists^{\geq 2}, \ldots\}$ and contains some $\exists^{\geq j}$ for $j \geq 2$ then X-CSP(Γ) displays **dichotomy** on the class of equality languages Γ:*

Theorem	Subsets
1	$\emptyset$
2	$\{\forall^{>2}\}$, $\{\forall^{>1}, \forall^{>2}\}$
3	$\{\forall^{>1}\}$
4	$\{\exists^{\geq 2}\}$
5	$\{\forall^{>2}, \exists^{\geq 2}\}$, $\{\forall^{>1}, \forall^{>2}, \exists^{\geq 2}\}$
6	$\{\forall^{>1}, \exists^{\geq 2}\}$

Figure 1: Classification theorems linked to canonical subsets of $\{\exists^{\geq 2}, \forall^{>1}, \forall^{>2}\}$

- *if all relations of* Γ *are* ***negative****, then* X*-CSP*(Γ) *is in L.*
- ***otherwise*** X*-CSP*(Γ) *is co-NP-hard.*

Theorem 5. *If* $\{\exists^{\geq 1}, \forall^{>0}\} \subseteq X \subseteq \{\forall^{>0}, \forall^{>1}, \forall^{>2}, \ldots, \exists^{\geq 1}, \exists^{\geq 2}, \ldots\}$ *and contains some* $\exists^{\geq i}$ *and* $\forall^{>j}$ *for* $i, j \geq 2$ *then* X*-CSP*(Γ) *displays* ***dichotomy*** *on the class of equality languages* Γ*:*

- *if all relations of* Γ *are* ***trivial****, then* X*-CSP*(Γ) *is in L.*
- ***otherwise*** X*-CSP*(Γ) *is Pspace-complete.*

Theorem 6. *If* $\{\exists^{\geq 1}, \forall^{>0}\} \subseteq X \subseteq \{\forall^{>0}, \forall^{>1}, \exists^{\geq 1}, \exists^{\geq 2}, \ldots\}$ *and contains* $\forall^{>1}$ *and some* $\exists^{\geq i}$ *for* $i \geq 2$ *then* X*-CSP*(Γ) *displays* ***dichotomy*** *on the class of equality languages* Γ*:*

- *if* Γ *is* ***specially negative****, then* X*-CSP*(Γ) *is in P.*
- ***otherwise*** X*-CSP*(Γ) *is Pspace-complete.*

Four of our five new worlds are somewhat more conducive to analysis than that of [11], in that in them we have no gap across co-NP-hardness and Pspace-completeness. For the remaining world of Theorem 4, we are able to demonstrate that improving co-NP- to Pspace-hardness is likely to be as difficult as in Theorem 1. Indeed, from this it follows that our working hypothesis promotes co-NP-hardness to Pspace-hardness for Theorem 4 as well as Theorem 1.

Some of our results are not especially complicated and stem from simple manipulations rather than deep technical nous. Against this we set the nice

aesthetic of our results and the way in which they complement [11] and [21]. For example, the specially negative languages, play an important role in our classifications, but where do they sit in the context of [21]? Do they even form a class of relations closed under pp-definitions (a so-called *co-clone*)? We note that equality languages have been also studied for the abduction problem [22], in the context of which one finds trichotomy among Σ_2^{P}-complete, NP-complete and problems decidable in P.

The paper is organised as follows. After the preliminaries, we address in Section 3 basic upper and lower bounds that play a role in our classification. In Section 4 we describe the crucial specially negative languages. Finally, we ponder the difficulty of improving, in Theorem 4, co-NP- to Pspace-hardness. We do the latter by showing that $\mathrm{QCSP}(x = y \vee u = v, \neq)$ becomes no more complex when one augments with $\exists^{\geq k}$.

2. Preliminaries

Let $[j] := \{1, \ldots, j\}$. For $i \in \mathbb{N}$, $i \geq 1$, let $\exists^{\geq i} x$ quantify that there exist at least i elements satisfying some property. For $i \in \mathbb{N}$, $i \geq 0$, let $\forall^{>i} x$ quantify that for all but at most i elements does some property hold (in the conference version of this paper the latter was $\forall^{\geq i}$ but it seems $\forall^{>i}$ is pedagogically better). Thus $\exists$ is $\exists^{\geq 1}$ and $\forall$ is $\forall^{>0}$. In this paper we consider languages with first-order (fo) definitions in equality, that is structures which admit all permutations as automorphisms (for a discussion of this equivalence see [9]). Such an *equality language* may be considered a structure of the form $(\mathbb{N}; R_1, \ldots)$ where each R_i is a CNF formula whose atoms are equalities or disequalities (owing to quantifier elimination of $\forall$ and $\exists$ this is equivalent to our saying fo-definable in equality). We typically drop the "$\mathbb{N}$" in referring to an equality language, indeed any infinite set could equivalently take its place. Note that we tend to conflate constraint languages and relational structures, preferring to deal concretely with the latter.

Primitive positive (pp) logic is the restriction of fo-logic to the symbols $\{\exists, \wedge, =\}$ and *positive Horn* (pH) likewise to the symbols $\{\forall, \exists, \wedge, =\}$. That is positive Horn logic is primitive positive logic with the universal quantifier restored. Without loss of generality we may assume formulas of these logics are in prenex normal form. For $X \subseteq \{\forall^{>0}, \forall^{>1}, \ldots, \exists^{\geq 1}, \exists^{\geq 2}, \ldots\}$, let X-pp denote the logic of prenex sentences whose symbols are among $X \cup \{\wedge, =\}$. We will use small letters such as ϕ to refer to sentences whose quantifier-free part will be denoted Φ. Let Γ be a set of relations on a domain, i.e. a

structure, and let $\mathscr{L}$ be a logic. Then the *evaluation problem for* $\mathscr{L}$ *on* Γ has as input a sentence $\phi \in \mathscr{L}$ and asks whether $\Gamma \models \phi$? The evaluation problem for primitive positive (resp., positive Horn) logic on Γ is better known as CSP(Γ) (resp., QCSP(Γ)). The evaluation problem for X-pp on Γ will henceforth be known as X-CSP(Γ). In this paper we consider only $X \supseteq \{\exists^{\geq 1}, \forall^{>0}\}$, i.e. extensions of the QCSP.

Let Γ be an equality language, and let $\langle\Gamma\rangle_{\text{pp}}$ be the set of relations pp-definable over Γ. Such a set is termed a *co-clone*. We may abuse notation and write $\langle R\rangle_{\text{pp}}$ when properly we mean $\langle\{R\}\rangle_{\text{pp}}$. A great project was launched in [21] to identify sets of the form $\langle\Gamma\rangle_{\text{pp}}$, charting their inclusion relations in a lattice. The lattice is mostly identified through a dually-isomorphic algebraic lattice of *local clones*, but here we will only be interested in the co-clones. In line with [21] and [11], we will consider the co-clone $\langle\emptyset\rangle_{\text{pp}} = \langle=\rangle_{\text{pp}}$ to be at the top of the lattice, with the co-clone of all equality-definable relations at the bottom.

In a problem X-CSP(Γ) it is desirable that Γ involves only a finite number of relations lest there arise the question as to how they are encoded. Yet we will enjoy referring to co-clones that contain an infinite number of relations. We resolve this by considering all references to Γ as a language inside X-CSP(Γ) to be restricting of Γ to the relationally finite.[1]

We now recall some basic results from [11, 21]. We typically write $\neq$ to indicate the binary relation of disequality (in terms of our notation this abbreviates $x \neq y$). Let $\langle\Gamma\rangle_{\text{pH}}$ be the set of relations pH-definable over Γ (indeed, let $\langle\Gamma\rangle_{X\text{-pp}}$ be the set of relations X-pp-definable over Γ). We will borrow the following result from [23].

Lemma 1 (Lemma 5.2 in [23]). $(x = y \vee u = v) \in \langle\Gamma\rangle_{\text{pp}}$ *iff* $\langle\Gamma\rangle_{\text{pp}}$ *is closed under definitions in existential positive first-order logic.*

Lemma 2.

(1.) The three co-clones $\langle x = y \vee u = v\rangle_{\text{pp}}$, $\langle x = y \vee y = z\rangle_{\text{pp}}$ *and* $\langle x = y \vee y = z \vee x = z\rangle_{\text{pp}}$ *coincide. That is,* $x = y \vee u = v$ *is in all three.*

(2.) $x \neq y \vee u \neq v \in \langle x = y \to y = z, \neq\rangle_{\text{pp}}$.

[1] Another typical solution is to give definition to complexity of relationally-infinite Γ along the lines of "easy", if it is easy for all finite subsets, and "hard", if it is hard for some finite subset.

(3.) $(x_1 = y \wedge \ldots \wedge x_m = y) \to y = z \in \langle x = y \to y = z \rangle_{\mathrm{pp}}$.

(4.) $\langle \{x = y \vee u = v, \neq\} \rangle_{\mathrm{pp}}$ *contains all equality-definable relations.*

(5.) $\langle x = y \vee u = v \rangle_{\mathrm{pp}}$ *contains all positive relations.*

(6.) If Γ *is not positive then* $\neq \in \langle \Gamma \rangle_{\mathrm{pp}}$.

(7.) If Γ *is positive but not negative then* $x = y \vee u = v \in \langle \Gamma \rangle_{\mathrm{pH}}$.

Proof. Notes. 1.) See Lemma 5.3.2 of [10]. The point is that $x = y \vee u = v$ and $x = y \vee y = z$ and $x = y \vee y = z \vee x = z$ have precisely the same (so-called) *polymorphisms*, i.e. all essentially unary operations. 2.) This appeared in an early unpublished version of [11] (intermediate between conference and journal), the proof of which we reproduce. Define $a \neq b \vee c \neq d$ as

$$\exists b', d', t \; a = b \to b = b' \wedge b = b' \to b' = t \wedge c = d \to d = d' \wedge d = d' \to d' = t \wedge b' \neq d'.$$

3.) Follows from [21]; for an explicit construction use:

$$\exists p_1, \ldots, p_m \; \left(\bigwedge_{i \in [m]} x_i = y \to y = p_i \right) \wedge \; \left(\bigwedge_{i \in [m-2]} p_i = p_{i+1} \to p_{i+1} = p_{i+2} \right) \wedge p_{m-1} = p_m \to p_m = z.$$

4.) This is a consequence of Lemma 1 since the existential positive closure of $\{x = y \vee u = v, \neq\}$ is plainly equal to the full closure of $\{x = y \vee u = v\}$ under (quantifier-free) logic.

5.) See Theorem 8 Part 1 in [21]. It is also a consequence of Lemma 1 since the existental positive closure of $\{x = y \vee u = v\}$ includes all definitions in positive (quantifier-free) logic.

6.) See Proposition 7.3 in [11]. 7.) See proof of Theorem 7.1 in [11]. The point is that $x = y \vee y = z$ will be pH-definable, implying the same for $x = y \vee u = v$, as in Part 1 above.

Note that Part 7 does not hold for pp-definability. While $x = y \vee y = z$ is a relational basis (under pp-definitions) for the positive languages, there is an infinite chain of positive languages up to pp-closure [21]. $\langle x = y \vee y = z \rangle_{\mathrm{pp}}$ is at the bottom, being the most expressive, and the trivial $\langle = \rangle_{\mathrm{pp}}$ is at the top. In between, is an infinite chain without top element. However, this chain collapses for pH-definability, leaving only the two co-clones up to pH-closure ($\langle = \rangle_{\mathrm{pp}}$ and $\langle x = y \vee y = z \rangle_{\mathrm{pp}}$).

We contrast Part 1 of Lemma 2 with the knowledge that $x = y \to u = v$ is in neither $\langle x = y \to y = z \rangle_{\mathrm{pp}}$ [21] nor $\langle x = y \to y = z \rangle_{\mathrm{pH}}$.

Owing to the disparity between the conference and journal versions of [11], we give the following as a specific proposition. Its proof can be derived from [11] by the assiduous reader.

Proposition 1. *Both QCSP*$(x = y \vee u = v, \neq)$ *and QCSP*$(w = z_1 \vee w = z_2 \vee w = z_3, \neq)$ *are Pspace-complete.*

Proof. QCSPs over equality languages are always in Pspace (see, e.g., [11], or look forward to Lemma 4). Let $\mathcal{K}_3$ be the irreflexive 3-clique (triangle graph). We reduce from QCSP($\mathcal{K}_3$), a.k.a. Quantified-3-Colouring, known to be Pspace-complete from [24]. Initially we will prove that QCSP$(w = z_1 \vee w = z_2 \vee w = z_3, \neq)$ is Pspace-hard. For an input ψ to Quantified-3-Colouring we build an instance ϕ of QCSP$(w = z_1 \vee w = z_2 \vee w = z_3, \neq)$ as follows. ϕ begins quantified outermost with $\exists c_1, c_2, c_3$, with $c_1 \neq c_2 \wedge c_1 \neq c_3 \wedge c_2 \neq c_3$ in its quantifier-free part (representing the three distinct colours). Now the quantifiers progress inwards in ϕ exactly as they did in ψ; and each atom $E(x, y)$ in ψ, becomes $x \neq y$ in ϕ. Further, for each existential variable x of ψ we add the conjunct $(x = c_1 \vee x = c_2 \vee x = c_3)$ (we do not need to do anything special for universal variables). This reduction, working by local replacement, can plainly be done in logarithmic space. Its correctness is straightforward, and the result for QCSP$(w = x \vee w = y \vee w = z, \neq)$ follows.

To obtain the result for QCSP$(x = y \vee u = v, \neq)$ it is sufficient to observe that $w = z_1 \vee w = z_2 \vee w = z_3$ is in $\langle \{x = y \vee u = v, \neq\} \rangle_{\mathrm{pp}}$, as guaranteed by Lemma 2 Part 4, and witnessed by the pp-definition

$$\exists c_1, c_2, c_3\ (w = c_1 \vee w = z_1) \wedge (w = c_2 \vee w = z_2) \wedge (w = c_3 \vee w = z_3) \wedge (c_1 \neq c_2 \vee c_2 \neq c_3).$$

To see that this works, note that $z_1 \neq z_2 \vee z_2 \neq z_3$ is equivalent to $z_1 \neq z_2 \vee z_2 \neq z_3 \vee z_1 \neq z_3$, and note that it is itself technically a shorthand for $\exists z'_1, z'_3\ z'_1 \neq z_1 \wedge z'_3 \neq z_3 \wedge (z'_1 = z_2 \vee z_2 = z'_3)$.

It seems that QCSP$(x = y \to u = v)$ is also Pspace-complete [13], but this is harder work. The backbone of a proof appeared in an early unpublished version of [11]. The first author has verified this proof but reproducing it here is beyond our scope.

3. Upper and lower bounds

3.1. Upper bounds

The following lemma is trivial but we state it because we will wish to appeal to it in the future.

Lemma 3 (Substitution of equalities). *Let* $X := \{\forall^{>0}, \forall^{>1}, \exists^{\geq 1}, \exists^{\geq 2}, \ldots\}$ *and let* Φ *be an instance of some* X*-CSP containing an equality* $x = y$ *in which* y *appears later in the quantifier order of* Φ *than* x*. Then* Φ *is false if* y *is quantified by anything other than* $\exists^{\geq 1}$*. Otherwise,* Φ *is equivalent to* Φ' *obtained by substituting all instances of* y *by* x *and removing the quantifier* $\exists^{\geq 1} y$*.*

Lemma 4. *For any* $X \subseteq \{\forall^{>0}, \forall^{>1}, \ldots, \exists^{\geq 1}, \exists^{\geq 2}, \ldots\}$*,* X*-CSP*(Γ) *is in Pspace.*

Proof. Let ϕ be an input sentence of X-pp containing n variables. It is clear that on an equality language we may substitute quantifiers $\exists^{\geq m}$, $m > n$ (resp., $\forall^{>m}$, $m > n$), by $\exists^{\geq n}$ (resp., $\forall^{>n}$), without affecting the truth of the sentence. Now we may evaluate by a simple branch and backtrack algorithm. The key point is that when the algorithm is about to branch on a variable, then it only needs to consider the valuations of the previous variables, together with at most **one** new value that is different. This is because equality languages have all permutations as automorphisms.

Note that Lemma 4 follows also from the result that the first-order theory of equality is Pspace-complete [25], though one needs to encode $\forall^{>i}$ and $\exists^{\geq i}$ carefully to avoid exponential blow-up.

The following lemma relates to $\{\exists^{\geq 1}, \forall^{>i} : i \geq 0\}$ on positive languages.

Lemma 5. *Let* $X := \{\exists^{\geq 1}, \forall^{>i} : i \geq 0\}$ *and* Γ *be a positive equality language. Then* X*-CSP*(Γ) *is in NP.*

Proof. Let n be the number of variables in the input X-pp sentence ϕ. We may substitute quantifiers $\forall^{>m}$, $m > n$, by $\forall^{>n}$, without affecting the truth of the sentence. In the following $Q\overline{x}$ is intended to signify a sequence of variables quantified by elements of the set X. We now eliminate quantification on y of the form $Q\overline{x}\ \forall^{>i} y\ Q\overline{z}\ \Phi(\overline{x}, y, \overline{z})$ by replacing it with $Q\overline{x}\ \exists y^1, \ldots, y^i \forall y\ Q\overline{z}\ y = y^1 \vee \ldots \vee y = y^i \vee \Phi(\overline{x}, y, \overline{z})$. It is known that positive sentences may be evaluated on equality languages in NP from [26], and the result follows.

3.2. Lower bounds

The following proposition relates to $\{\forall^{>i}, \forall^{>0}, \exists^{\geq 1}\}$, for $i \geq 2$, on non-positive languages.

Proposition 2. *Let $X := \{\forall^{>i}, \forall^{>0}, \exists^{\geq 1}\}$, for any $i \geq 2$ and Γ a non-positive equality language. Then X-CSP(Γ) is Pspace-complete.*

Proof. Since Γ is non-positive, $\neq \in \langle\Gamma\rangle_{\text{pp}}$ by Lemma 2 Part 6.

Let $i \geq 2$ be fixed. Note that $\forall^{>i} z\ z \neq z_1 \wedge \ldots \wedge z \neq z_{i+1}$ defines $\bigvee_{\lambda \neq \mu \in [i]} z_\lambda = z_\mu$. Consequently, from Lemma 2, Parts 1 and 4, $\forall^{>i}$ and $\exists$ define all equality relations on $\neq$, and Pspace-hardness follows from Proposition 1.

The following propositions relate to $\{\forall^{>1}, \forall^{>0}, \exists^{\geq 1}\}$ on non-positive languages.

Proposition 3. $\{\forall^{>1}, \forall^{>0}, \exists^{\geq 1}\}$*-CSP($x \neq y \vee u \neq v$) is Pspace-complete.*

Proof. Clearly we can (pp-)define $\neq$. Further, we can define, now permitting also $\forall^{>1}$, $(x = y \vee u = v)$ as

$$\forall^{>1} p\ \forall^{>1} q\ (x \neq p \wedge p \neq y) \vee (u \neq q \wedge q \neq v),$$

which expands to make:

$$\forall^{>1} p\ \forall^{>1} q\ (x \neq p \vee u \neq q) \wedge (x \neq p \vee q \neq v) \wedge (p \neq y \vee u \neq q) \wedge (p \neq y \vee q \neq v).$$

The result now follows from the Pspace-hardness of QCSP($\{x = y \vee u = v, \neq\}$) as per Proposition 1.

Proposition 4. *$(x \neq y \vee u \neq v)$ is definable from $(x \neq y \vee y \neq z)$ using $\forall^{>1}$.*

Proof. We will give a definition of $(x = y \vee y \neq z)$, whereupon we may appeal to Lemma 2 Part 2. Firstly, we define $x \neq y \vee u \neq v \vee x = y = u = v$ by

$$\forall^{>1} p\ (p = x \to x \neq y) \wedge (u = v \to v \neq p).$$

Now $(x = y \vee y \neq z)$ is equivalent to $\forall^{>1} p\ (x \neq p \wedge p \neq y) \vee y \neq z$ which is clearly $\forall^{>1} p\ (x \neq p \vee y \neq z) \wedge (p \neq y \vee y \neq z)$, but also is equivalent to $\forall^{>1} p\ (x \neq p \vee y \neq z \vee x = p = y = z) \wedge (p \neq y \vee y \neq z)$, as $x = p = y = z$ violates the second clause.

Proposition 5. $\{\forall^{>1}, \forall^{>0}, \exists^{\geq 1}\}$-*CSP*$(x \neq y \vee y \neq z)$ *is Pspace-complete.*

Proof. Follows directly from the previous two propositions.

Proposition 6. *Let* $i \geq 2$ *and* Γ *be a positive non-trivial equality language. Then* $\neq$ *is definable from* Γ *using* $\exists^{\geq i}$.

Proof. First, see that $(x = y \vee u = v) \in \langle\Gamma\rangle_{\mathrm{pH}}$ from Lemma 2 Part 7 (recalling positive non-trivial is positive non-negative). Now we note that, for each $j \geq 2$, $(u = v_1 \vee \ldots \vee u = v_j)$ is in $\langle x = y \vee y = z\rangle_{\mathrm{pp}}$, via:

$$\begin{aligned}\exists p_1, \ldots, p_{j-2} \quad & (u = v_1 \vee u = p_1) \wedge (p_1 = v_2 \vee p_1 = p_2) \wedge \ldots \\ & \wedge (p_{j-3} = v_{j-2} \vee p_{j-3} = p_{j-2}) \wedge (p_{j-2} = v_{j-1} \vee p_{j-2} = v_j).\end{aligned}$$

Finally, we let $i = j$ and say $\exists^{\geq j} u\ (u = v_1 \vee \ldots \vee u = v_j)$ which is equivalent to $\bigwedge_{\lambda \neq \mu \in [j]} v_\lambda \neq v_\mu$. We now obtain $v_1 \neq v_2$ by existentially quantifying $v_3, \ldots, v_k$.

4. Isolating the specially negative languages

We call a CNF Φ *reduced* if it is not logically equivalent to itself with either a clause or a literal in a clause removed. A CNF *depends* on one of its variables v if its truth value can not be given as a propositional function of (the equality type of) only its other variables. For example, the single-clause CNF $(x \neq y \vee y \neq z \vee x \neq z)$ is not reduced since it is equivalent to each of the reduced CNFs $(x \neq y \vee y \neq z)$, $(x \neq y \vee x \neq z)$ and $(y \neq z \vee x \neq z)$. This reminds us that reduced CNFs need not appear uniquely in (unreduced) CNFs. Meanwhile, $x \neq y \wedge (y = z \vee y \neq z)$ depends on x and y, but not on z.

Suppose R is a negative relation which might be given by various reduced CNFs, at least one of which, Φ, is negative. Then Φ may enforce some equalities on its variables, which it plainly does syntactically. Sometimes in this section we will wish to assume that Φ has these equalities factored out by substitution, thus we could assume negative CNFs do not have any positive clauses. The point is that this is an innocuous assumption. Having said that, the process does remove variables and so cannot be used strictly within a quantifier elimination procedure. Now, suppose R is also given by another reduced CNF Φ' which is not negative (we will later prove this is not possible). Then we could similarly factor out the implied equalities of Φ', but these might not be obvious unless we have negative Φ where the equalities are syntactically explicit and not semantically implied.

4.1. Non-positive cases involving $\forall^{>1}$.

Definition 1. *A negative, reduced CNF Φ without equalities is flat if it consists of clauses with no variable occurring twice in them. Φ is rich if every variable of Φ occurs in a singleton clause of Φ.*

We will assume that CNFs do not possess dummy variables that do not appear explicitly. Thus, we may consider a CNF to be rich if it is rich once we have discounted such dummy variables.

Example 1. *The formula $(x \neq y \vee u \neq v) \wedge x \neq z \wedge (x \neq u \vee y \neq v)$ is flat and the formula $(x \neq y \vee u \neq v) \wedge x \neq u \wedge y \neq v$ is rich.*

Theorem 7. *We have the following dichotomy for non-positive equality languages Γ.*

1. *Either $\langle\Gamma\rangle_{\{\exists,\forall,\forall^{>1}\}\text{-pp}}$ contains the relation $x \neq y \vee y \neq z$, and then we find $\{\exists, \forall, \forall^{>1}\}$-CSP$(\Gamma)$ is Pspace-hard,*
2. *or $\langle\Gamma\rangle_{\{\exists,\exists^{\geq 2},\ldots,\forall,\forall^{>1}\}\text{-pp}}$ contains only relations whose reduced CNFs are negative, flat and rich; and $\{\exists, \exists^{\geq 2}, \ldots, \forall, \forall^{>1}\}$-CSP$(\Gamma)$ is in P.*

The proof of this theorem will come at the end of this section. Note the discrepancy of the co-clones in the two parts with $\exists^{\geq j}$ $(j > 1)$ appearing in the latter but not the former.

Definition 2. *Let Φ be a CNF with variables in V. Let $P_1 \cup \cdots \cup P_k$ be a partition of V. We say that "we weaken Φ around the given partition by keeping $P_1, \ldots, P_j$" $(j \leq k)$ if we produce a formula from Φ by the following definition. First we take the conjunction Ψ of all disequalities that are transversal to the partition, i.e., $x \neq y$ with x and y not in the same set of the partition, and produce $\Phi \wedge \Psi$. Then for all $i > j$ we identify the variables in P_i by a new variable w_i. Then we existentially quantify over these new variables.*

Note that if Φ pp-defines $\neq$, then this procedure yields a pp-definition of $\Phi \wedge \Psi$ from Φ.

Example 2. *We give the following example for weakening $\Phi := (x \neq y \vee u \neq v \vee s \neq t) \wedge (u \neq s \vee v \neq t)$ around the partition $P_1 = \{x, y, u, v\}, P_2 = \{s, t\}$ by keeping P_1 (i.e. $j = 1$). Then $\Phi \wedge \Psi$ is*

$$(x \neq y \vee u \neq v \vee s \neq t) \wedge (u \neq s \vee v \neq t) \wedge$$
$$x \neq s \wedge x \neq t \wedge y \neq s \wedge y \neq t \wedge u \neq s \wedge u \neq t \wedge v \neq s \wedge v \neq t,$$

which is logically equivalent to

$$(x \neq y \vee u \neq v \vee s \neq t) \wedge x \neq s \wedge x \neq t \wedge y \neq s \wedge y \neq t \wedge u \neq s \wedge u \neq t \wedge v \neq s \wedge v \neq t,$$

which after identification and quantification collapses to $x \neq y \vee u \neq v$ (cf. Lemma 9).

Lemma 6. *Let Φ be a reduced negative CNF that has a non-flat clause. Then Φ pp-defines $x \neq y \vee y \neq z$.*

Proof. First we recall that Φ pp-defines $\neq$. We can partition the free variables V of Ψ into $P_1 \cup \cdots \cup P_k$, for each negative clause C, so that the equivalence relation with classes $P_1, \ldots, P_k$ is the symmetric transitive closure of the graph with edges $(x, y) \in V^2$ such that $x \neq y$ is a literal in C. Then the clause C expresses that there are two equivalent variables (i.e., in the same set of the partition) whose values are not equal. Let $\Phi \equiv C_1 \wedge \cdots \wedge C_r$. Let C_1 be a non-flat clause. We weaken Φ around the partition corresponding to C_1 by keeping an at least 3-element set in the partition (which exists because the clause is not flat). So let Ψ be the conjunction of all disequalities $x \neq y$ such that x and y are in different sets of the partition corresponding to C_1. Then $\Phi \wedge \Psi$ is logically equivalent to $C_1 \wedge \Psi$. Indeed, as Φ is reduced, there cannot be two clauses in Φ such that the partition corresponding to one is finer than the other, so for all C_i with $i \neq 1$ there is some disequality in Ψ that is stronger than C_i. Let P be a set in the partition corresponding to C_1 with at least 3 elements. For all other sets S in the partition identify the variables in S with a single variable (using different variables for different sets). Quantify existentially over these new variables in the formula $\Phi \wedge \Psi$. We obtain a formula Φ' that is logically equivalent to $x_1 \neq x_2 \vee x_2 \neq x_3 \cdots \vee x_{p-1} \neq x_p$ where $p = |P| \geq 3$.

By applying the identifications $x = x_1$, $y = x_2$, $z = x_3 = \cdots = x_p$ we obtain the formula $x \neq y \vee y \neq z$.

Lemma 7. *From $S(x, y, u, v) := (x \neq y \vee u \neq v) \wedge y \neq u \wedge x \neq u$ we may define with $\forall^{>1}$ and $\exists$ the relation $p \neq q \vee q \neq r$.*

Proof. Consider $\forall^{>1} v\ S(x, y, u, v) \wedge S(x', y', u', v)$ which simplifies to

$$y \neq u \wedge u \neq x \wedge y' \neq u' \wedge u' \neq x' \wedge (u = u' \vee x \neq y \vee x' \neq y').$$

Now add outermost $\exists x', y'$ together with the atom $x' = y'$ to obtain the relation $x \neq u \wedge y \neq u \wedge (u = u' \vee x \neq y)$. Finally, we add outermost $\exists u$ to obtain $(u' \neq x \wedge u' \neq y) \vee x \neq y$ which is logically equivalent to, e.g., $(x \neq y \vee u' \neq x)$.

Lemma 8. *Let Φ be a flat negative CNF with exactly four variables. Assume that Φ depends on each of its variables, and that Φ is not rich. Then $x \neq y \vee y \neq z$ has an $\{\exists, \forall, \forall^{>1}\}$-pp definition in Φ.*

Proof. With a case-by-case analysis we can show that the formula Φ pp-defines $(x \neq y \vee u \neq v) \wedge y \neq u \wedge x \neq u$, and thus we are done by Lemma 7.

We may assume that $x \neq y \vee u \neq v$ is a clause of Φ, and v is not in a singleton clause. We have the following possibilities, up to exchanging variables.

1. $(x \neq y \vee u \neq v)$
2. $(x \neq y \vee u \neq v) \wedge y \neq u$
3. $(x \neq y \vee u \neq v) \wedge y \neq u \wedge x \neq u$
4. $(x \neq y \vee u \neq v) \wedge (x \neq u \vee y \neq v)$
5. $(x \neq y \vee u \neq v) \wedge (x \neq u \vee y \neq v) \wedge y \neq u$
6. $(x \neq y \vee u \neq v) \wedge (x \neq u \vee y \neq v) \wedge (x \neq v \vee y \neq u)$

Note that $\neq$ is pp-definable from Φ. Thus $\Phi \wedge y \neq u \wedge x \neq u$ is pp-definable from Φ, and it is logically equivalent to $(x \neq y \vee u \neq v) \wedge y \neq u \wedge x \neq u$.

Lemma 9. *Assume that $\langle\Gamma\rangle_{\{\exists,\forall,\forall^{>1}\}\text{-pp}}$ contains a negative CNF Φ that is not rich. Then $\langle\Gamma\rangle_{\{\exists,\forall,\forall^{>1}\}\text{-pp}}$ contains the formula $x \neq y \vee y \neq z$.*

Proof. We may assume that Φ is flat, as otherwise we are done by Lemma 6. Let x be a free variable of Φ that does not occur in a singleton clause. As x is a free variable, it must be in some clause of Φ; let C be the shortest one. By assumption, C has length at least two. Let y, u, v be free variables of Φ such that C is of the form $x \neq y \vee u \neq v \vee \cdots$. Consider the partition corresponding to C, and unify the two 2-element sets $\{x, y\}$, $\{u, v\}$ to obtain a coarser partition E with a single 4-element subset $P = \{x, y, u, v\}$. Recalling that Φ pp-defines $\neq$, let us weaken Φ around E by keeping P. After reducing the result, we obtain a formula Ψ with free variables $\{x, y, u, v\}$. Again, this must be flat, or we are done by Lemma 6. If Ψ is rich, then x is in a singleton clause of Ψ. This singleton clause must have come from a clause of Φ that was shorter than C, a contradiction. Hence, Ψ $\{\exists, \forall, \forall^{>1}\}$-pp defines $x \neq y \vee y \neq z$ by Lemma 8.

Note that the $\{\exists, \forall, \forall^{>1}\}$-pp definitions of Lemmas 8 and 9 did not need $\forall$.

4.2. Quantifier elimination and reduction for negative, flat and rich CNF formulas

We wish to argue that in a certain case we can effect quantifier elimination for negative, flat and rich CNF formulas in polynomial time. Eliminating a $\forall^{>1}$ quantifier over negative, flat and rich formulas can throw up Horn CNFs, so our first task will be to argue that we can compute a reduced form from a Horn CNF Φ in polynomial time. Our task is to determine, whether there are any redundant clauses (and if so remove them) and then whether there are any redundant literals in the remaining clauses (which also must be removed). To test for redundant clauses C it is sufficient to test whether $(\Phi \setminus \{C\})$ implies C, i.e. whether $(\Phi \setminus \{C\}) \wedge \neg C$ is a contradiction. But this is itself the complement of a Horn CSP, itself uniformly tractable by unit propagation (see [9])! Similarly, to determine if a literal ℓ is required in a clause C, we may consider whether $(\Phi \setminus \{C\}) \wedge \ell \wedge \neg(C \setminus \{\ell\})$ is a contradiction. Thus, given a Horn CNF Φ we can compute in polynomial time a reduced CNF (itself Horn) Φ' that is equivalent to Φ.

Consider each $\forall^{>1} x_i\ \Phi(x_1, \ldots, x_k)$, for $i \leq k$, where Φ is a negative, flat and rich CNF, without any equalities, where the singletons involving x_i are precisely $x_i \neq x'_{\lambda_1}$, $\ldots$, $x_i \neq x'_{\lambda_t}$. We can effect quantifier elimination in the following fashion. We consider separately non-singleton and singleton clauses. Substitute each non-singleton clause involving x_i, itself of the form, $(x_i \neq x'_{\mu_1} \vee x_{\mu_2} \neq x'_{\mu_2} \vee \ldots \vee x_{\mu_s} \neq x'_{\mu_s})$, where no variable is repeated, by $(x'_{\mu_1} = x'_{\lambda_1} = \cdots = x'_{\lambda_t} \vee x_{\mu_2} \neq x'_{\mu_2} \vee \ldots \vee x_{\mu_s} \neq x'_{\mu_s})$. Note that we will allow conjuncts of equalities as single relations in our clauses, to avoid having to break the clauses up. If we view this as a notational shorthand then we do not break the condition of Horn-ness. If x_i appeared in multiple singleton clauses (i.e. $t > 1$), then we need to add the equality $x_{\lambda_1} = \cdots = x_{\lambda_t}$ to the system. If x_i appeared in a single singleton clause then we can now simply remove it. Call the new CNF finally obtained $\tilde{\Phi}$. Plainly, $\tilde{\Phi}$ is logically equivalent to $\forall^{>1} x_i\ \Phi$ and $\tilde{\Phi}$ is Horn. We now apply our effective procedure to establish whether $\tilde{\Phi}$ is reduced and if not reduce it.

The question now naturally arises as to whether $\tilde{\Phi}$ is negative. We argue by the following lemma that it is enough to see whether our reduced CNF form has a non-singleton clause with an equality in it.

Lemma 10. *Let Φ be a reduced CNF representing a relation R in which there is a non-singleton clause that contains an equality. It is not possible that R is negative (i.e. has another reduced CNF that is negative).*

PROOF. In line with the discussion at the start of this section, we assume that all clauses of Φ', itself derived from Φ, are negative (eliminating equalities by substitution if necessary).

Φ has a clause C of the form

$$(x_1 = x'_1 \vee \ldots \vee x_k = x'_k \vee y_1 \neq y'_1 \vee \ldots \vee y_\ell \neq y'_\ell).$$

Using the disequalities of C as an edge relation on (all) the variables $\{x_1, x'_1, \ldots, x_k, x'_k, y_1, y'_1, \ldots, y_\ell, y'_\ell\}$ we obtain a graph $\mathcal{G}$. Suppose that there is a path from some x_i to x'_i in $\mathcal{G}$, then plainly C is a tautology and Φ' was not a reduced CNF.[2]

Now, Φ has a satisfying assignment with $x_1 = x'_1$, $x_2 \neq x'_2$, $\ldots$, $x_k \neq x'_k$ and $y_1 = y'_1$, $\ldots$, $y_\ell = y'_\ell$. Since all clauses of Φ' are negative, we can deduce that adhering to the equalities $x_1 = x'_1$, $y_1 = y'_1$, $\ldots$, $y_\ell = y'_\ell$, but for other than these equivalence classes setting everything distinct, satisfies Φ' (and maintains $x_2 \neq x'_2$, $\ldots$, $x_k \neq x'_k$ by the discussion in the previous paragraph). But, since Φ' is negative, our assignment remains satisfying if we now force the variables x_1 (plus those in its connected component in $\mathcal{G}$) and x'_1 (plus those in its connected component in $\mathcal{G}$) to have distinct values, and keep that these values are distinct from any we used elsewhere. Crucially, this valuation now invalidates C and this is a contradiction.

We will now consider the quantifier elimination of $\exists$, $\exists^{\geq 2}$, $\ldots$ etc. (note that quantification by $\forall$ on a negative, flat and rich formula will always leave it false). On negative formulas, with equalities removed by substitution, $\exists^{\geq 2}$, $\ldots$ etc. have the same power as $\exists$ and allow us to remove any clause in which the corresponding variable appears. This leaves the formula negative and we again have an effective method for reduction.

We now continue in this vein eliminating quantifiers and reducing CNFs. If we at any point produce a CNF whose reduced form is not negative, flat and rich then we know that Γ $\{\exists, \forall, \forall^{>1}\}$-pp defines $x \neq y \vee y \neq z$. We are now in a position to address Theorem 7.

[2]Here we may as well make the assumption that the empty conjunction $\top$ is a negative formula since then we do not have to worry if C is the only clause of Φ'.

PROOF (OF THEOREM 7). If Γ is not negative, then it follows from [12] that Γ pp-defines either $x = y \vee y = z$ or $x = y \to y = z$. Thus it follows from Parts 4 and 1 or 2, respectively, of Lemma 2, that $x \neq y \vee y \neq z$ is pp-definable (in the former case $\neq$ is also needed). We are now in the first case of the theorem by Proposition 5.

Now, assume that Γ is negative and $\langle\Gamma\rangle_{\{\exists,\forall,\forall^{>1}\}\text{-}pp}$ contains a formula that is not flat or not rich. Then Γ falls into the first case of the theorem by Lemmas 6 and 9 and Proposition 5.

Finally assume that $\langle\Gamma\rangle_{\{\exists,\exists^{\geq 2},\ldots,\forall,\forall^{>1}\}\text{-}pp}$ consists only of negative, flat, rich formulas (recall that on the negative languages $\exists$ and $\exists^{\geq j}$ are equipotent). The quantifier-elimination scheme we have explained above runs in polynomial time and always removes a clause from the CNF and thus terminates in a linear number of steps.

By way of example for Theorem 7, we note that $\langle\{(u \neq v \vee x \neq y) \wedge u \neq y \wedge v \neq x\}\rangle_{\{\exists,\forall,\forall^{>1}\}\text{-}pp}$ contains only negative, flat and rich formulas.

5. Proofs of the main theorems

Since we appeal to it several times in the forthcoming proofs, we reproduce the following nugget from [11]. Note that NP-membership here comes from [26]

Theorem 8 (Theorem 7.1 in [11]). *Let Γ be a positive equality language that is not negative. Then QCSP(Γ) is NP-complete.*

We now give the proofs of the main theorems, reproducing the statements of those theorems for the reader's convenience.

Theorem 2. If $\{\exists^{\geq 1}, \forall^{>0}\} \subseteq X \subseteq \{\exists^{\geq 1}, \forall^{>0}, \forall^{>1}, \ldots\}$ and contains some $\forall^{>j}$ for $j \geq 2$, then X-CSP(Γ) displays trichotomy on the class of equality languages Γ**:** If all relations of Γ are trivial, then X-CSP(Γ) is in L; if all relations of Γ are positive, but some relation is not trivial, then X-CSP(Γ) is NP-complete; and otherwise X-CSP(Γ) is Pspace-complete.

PROOF. If Γ is trivial, then X-CSP(Γ) is in L due to Lemma 3 and [27]. For positive languages, NP membership follows from Lemma 5. NP-hardness for the non-trivial positive languages follows from Theorem 8. Pspace-hardness for non-positive languages follows from Proposition 2.

Theorem 3. If $X = \{\exists^{\geq 1}, \forall^{>0}, \forall^{>1}\}$, then X-CSP(Γ) displays trichotomy on the class of equality languages Γ**:** If Γ is specially negative, then X-CSP(Γ) is in P; if all relations of Γ are positive, but some relation is not trivial, then X-CSP(Γ) is NP-complete; and otherwise X-CSP(Γ) is Pspace-complete.

Proof. For specially negative languages, P membership follows from Theorem 7. For positive languages, NP membership follows from Lemma 5. NP-hardness for the the non-trivial positive languages follows from Theorem 8. Pspace-hardness for the remaining languages follows from Theorem 7.

Theorem 4. If $\{\exists^{\geq 1}, \forall^{>0}\} \subseteq X \subseteq \{\forall^{>0}, \exists^{\geq 1}, \exists^{\geq 2}, \ldots\}$ and contains some $\exists^{\geq j}$ for $j \geq 2$ then X-CSP(Γ) displays dichotomy on the class of equality languages Γ**:** If all relations of Γ are negative, then X-CSP(Γ) is in L; and otherwise X-CSP(Γ) is co-NP-hard.

Proof. For negative languages, it is easy to see that $\exists^{\geq j} x$ holds iff $\exists^{\geq 1} x$ (for all j), once we have eliminated equalities by substitution as in Lemma 3. This is due to the numerous automorphisms of equality languages. The logspace membership of negative languages follows from Theorem 6.1 in [11]. For positive, non-negative languages, we appeal to Proposition 6 together with Theorem 8. For non-positive and non-negative languages, we again appeal to Theorem 5.5 in [11].

Theorem 5. If $\{\exists^{\geq 1}, \forall^{>0}\} \subseteq X \subseteq \{\forall^{>0}, \forall^{>1}, \ldots, \exists^{\geq 1}, \exists^{\geq 2}, \ldots\}$ and contains some $\exists^{\geq i}$ and $\forall^{>j}$ for some $i, j \geq 2$ then X-CSP(Γ) displays dichotomy on the class of equality languages Γ**:** If all relations of Γ are trivial, then X-CSP(Γ) is in L; and otherwise X-CSP(Γ) is Pspace-complete.

Proof. If Γ is trivial, then X-CSP(Γ) is in L due to Lemma 3 and [27]. Pspace-hardness follows from Proposition 2 via Proposition 6, the latter of which permits the definition of $\neq$.

Theorem 6. If $\{\exists^{\geq 1}, \forall^{>0}\} \subseteq X \subseteq \{\forall^{>0}, \forall^{>1}, \exists^{\geq 1}, \exists^{\geq 2}, \ldots\}$ and contains $\forall^{>1}$ and some $\exists^{\geq i}$ for some $i \geq 2$ then X-CSP(Γ) displays dichotomy on the class of equality languages Γ**:** If Γ is specially negative, then X-CSP(Γ) is in P; and otherwise X-CSP(Γ) is Pspace-complete.

Proof. Membership in P follows from Theorem 7. If Γ is positive non-trivial, then $\neq$ is definable by Proposition 6 and Pspace-hardness follows from Proposition 1. For negative languages that are not specially negative,

Pspace-hardness follows from Theorem 7. Finally, for Pspace-hardness of languages that are neither positive nor negative, we apply Lemma 2 Part 6 with Proposition 1.

6. Ennui of co-NP- to Pspace-completeness

We now ponder why it might be a challenge to improve any of our general co-NP-hardness results to Pspace-completeness.

Proposition 7. *For all* $1 \leq k \in \mathbb{N}$, $\{\exists^{\geq k}, \forall^{>0}, \exists^{\geq 1}\}$-*CSP*$(x = y \to y = z)$ *and* $\{\forall^{>0}, \exists^{\geq 1}\}$-*CSP*$(x = y \to y = z)$ *(i.e.* *QCSP*$(x = y \to y = z)$*) are logspace equivalent.*

Proof. The trivial identity reduction works in the backward direction, for the forward direction we will give a procedure to eliminate quantifiers of the form $\exists^{\geq k}$ from an input ϕ of the form $Q\overline{x}\ \Phi(\overline{x})$. Suppose Φ is constituted by the atoms $\Phi_1, \ldots, \Phi_r$, and ϕ contains s variables quantified by $\exists^{\geq k}$.

We eliminate quantification on y of the form $Q\overline{x}\ \exists^{\geq k} y\ Q\overline{z}\ \Phi(\overline{x}, y, \overline{z})$ by replacing it with

$$Q\overline{x}\ \exists y^1, \ldots, y^k \forall y\ Q\overline{z}\ \left(\bigwedge_{i \neq j \in [k]} y^i \neq y^j \right) \wedge (y = y^1 \vee \ldots \vee y = y^k) \to \Phi(\overline{x}, y, \overline{z}),$$

where $y^1, \ldots, y^k$ are new variables. In the quantifier-free part, because the antecedent $(y = y^1 \vee \ldots \vee y = y^k)$ is attained, we may consider $(y = y^1 \vee \ldots \vee y = y^k) \to \Phi(\overline{x}, y, \overline{z})$ to be replaced by a conjunction of atoms of the form $(y = y^1 \vee \ldots \vee y = y^k) \to \Phi_i$, if Φ_i involves y, *and just* Φ_i *otherwise.* Iterating this procedure, we can eliminate all quantifiers of the form $\exists^{\geq k}$ at the cost of introducing ks new variables. Each atom Φ_i of Φ contains at most three variables, so the iterated procedure can only cause it to be preceded *by at most three instances* of "$(y = y^1 \vee \ldots \vee y = y^k) \to$". This boundedness is essential to the working of the reduction.

We now address the removal of these at most three antecedents at the atomic level. That is, we show how to manipulate the atoms so that they are only of the form $x' = y' \to y' = z'$. It will be simplest to give this as an iterative procedure. If this procedure were to be applied a linear number of times, it would generate an exponential size blow-up, but we will only need to apply it three times and hence we avoid this. We need to consider

three cases, for the three possible variables in each Φ_i, being of the form $x = y \to y = z$.

Case 1. $(x = x^1 \vee \ldots \vee x = x^k) \to (x = y \to y = z)$. We will introduce two new variables p and q, adding innermost quantification $\exists p, q$. Add atoms $x = x^1 \to x = p, \ldots, x = x^k \to x = p$. We will also need $(x = p \wedge x = y) \to x = q$ and $y = q \to y = z$. That the former can be (pp-)expressed follows from Lemma 2 Part 3.

Case 2. $(y = y^1 \vee \ldots \vee y = y^k) \to (x = y \to y = z)$. We will introduce a new variable p, adding innermost quantification $\exists p$. Add atoms $y = y^1 \to y = p, \ldots, y = y^k \to y = p$. We will also need $(y = p \wedge y = x) \to y = z$. That this can be (pp-)expressed follows from Lemma 2 Part 3.

Case 3. $(z = z^1 \vee \ldots \vee z = z^k) \to (x = y \to y = z)$. We will introduce two new variables p and q, adding innermost quantification $\exists p, q$. Add atoms $z = z^1 \to z = p, \ldots, z = z^k \to z = p$. We will now also need $x = y \to y = q$ and $x = q \to q = p$.

It follows from Proposition 7 and [11] that our working conjecture that QCSP $(x = y \to y = z)$ is Pspace-complete would elevate the co-NP-hardness cases of Theorems 1 and 4 to Pspace-hardness.

7. Final remarks

We have classified the extensions of the quantified constraint satisfaction problem over equality languages in which the native existential and universal quantifiers are augmented by some subset of counting quantifiers. This is the first complexity classification for counting quantifiers over an infinite domain and the first to consider quantifiers of the form "for all but j". We have additionally built upon the work of [11] and [21]. The classes of equality languages designated positive and negative (also trivial) are not new, and appeared in those papers. However, the class of specially negative languages, which plays a role in our classification, is new and interesting. We are not sure exactly where this class fits within the negative languages or even if it forms a co-clone (is there more than one maximal specially negative language?).

Whilst Theorems 1 and 4 are complete classifications, in the sense of [11], they are incomplete in that many of the co-NP-hard cases are known to be Pspace-complete. It is here that the working hypothesis allows for a fuller classification and this that gives it a raison d'être.

Finally, it is appropriate to comment on our working hypothesis. Most people who have investigated the problem have begun with the strong belief that QCSP($x = y \to y = z, \neq$) is Pspace-complete. However, with time invested, this belief typically fades and one entertains also the possibility of co-NP-membership. For this reason we resist to phrase it as a conjecture per se.

Acknowledgements

We are grateful for many helpful corrections from reviewers for the journal version of this article.

[1] P. G. Kolaitis, M. Y. Vardi, Finite Model Theory and Its Applications (Texts in Theoretical Computer Science. An EATCS Series), Springer-Verlag New York, Inc., 2005, Ch. A logical Approach to Constraint Satisfaction.

[2] T. Feder, M. Y. Vardi, The computational structure of monotone monadic SNP and constraint satisfaction: A study through Datalog and group theory, SIAM Journal on Computing 28 (1999) 57–104. A preliminary version appeared in the proceedings of STOC'93.

[3] A. A. Bulatov, P. Jeavons, A. A. Krokhin, Classifying the complexity of constraints using finite algebras, SIAM J. Comput. 34 (3) (2005) 720–742.

[4] T. J. Schaefer, The complexity of satisfiability problems, in: Proceedings of STOC'78, 1978, pp. 216–226.

[5] A. Bulatov, A dichotomy theorem for constraint satisfaction problems on a 3-element set, J. ACM 53 (1) (2006) 66–120.

[6] P. Hell, J. Nešetřil, On the complexity of H-coloring, Journal of Combinatorial Theory, Series B 48 (1) (1990) 92–110.

[7] L. Barto, M. Kozik, T. Niven, The CSP dichotomy holds for digraphs with no sources and no sinks (a positive answer to a conjecture of Bang-Jensen and Hell), SIAM Journal on Computing 38 (5) (2009) 1782–1802. A preliminary version appeared in the proceedings of STOC'10.

[8] U. Egly, T. Eiter, H. Tompits, S. Woltran, Solving advanced reasoning tasks using quantified boolean formulas, in: Proc. 17th Nat. Conf. on Artificial Intelligence and 12th Conf. on Innovative Applications of Artificial Intelligence, AAAI Press/ The MIT Press, 2000, pp. 417–422.

[9] M. Bodirsky, J. Kára, The complexity of equality constraint languages, Theory of Computing Systems 43 (2) (2008) 136–158, a preliminary version appeared in the proceedings of CSR'06.

[10] M. Bodirsky, Complexity classification in infinite-domain constraint satisfaction, mémoire d'habilitation à diriger des recherches, Université Diderot – Paris 7. Available at arXiv:1201.0856 (2012).

[11] M. Bodirsky, H. Chen, Quantified equality constraints, SIAM J. Comput. 39 (8) (2010) 3682–3699.

[12] M. Bodirsky, H. Chen, Quantified equality constraints, in: Proceedings of LICS'07, 2007, pp. 203–212.

[13] M. Bodirsky, H. Chen, Personal communication (2012).

[14] B. Martin, F. R. Madelaine, J. Stacho, Constraint satisfaction with counting quantifiers, SIAM J. Discrete Math. 29 (2) (2015) 1065–1113. Extended abstracts appeared at CSR 2012 and CSR 2014.

[15] H.-D. Ebbinghaus, J. Flum, Finite Model Theory, Springer, 1999, 2nd edition.

[16] M. Otto, Bounded variable logics and counting – A study in finite models, Vol. 9, Springer-Verlag, 1997, iX+183 pages.

[17] C. S. Peirce (Ed.), Studies in Logic; by Members of the Johns Hopkins University (1883), John Benjamins Publishing, 1983.

[18] R. Clark, M. Grossman, Number sense and quantifier interpretation, Topoi 26 (1) (2007) 51–62.

[19] J. Szymanik, Quantifiers and Cognition: Logical and Computational Perspectives, Studies in Linguistics and Philosophy, Springer, 2016.

[20] D. A. M. Barrington, N. Immerman, H. Straubing, On uniformity within NC^1, J. Comput. Syst. Sci. 41 (3) (1990) 274–306.

[21] M. Bodirsky, H. Chen, M. Pinsker, The reducts of equality up to primitive positive interdefinability, J. Symb. Log. 75 (4) (2010) 1249–1292.

[22] J. Schmidt, M. Wrona, The Complexity of Abduction for Equality Constraint Languages, in: Computer Science Logic (CSL 2013), 2013, pp. 615–633.

[23] M. Bodirsky, M. Hils, B. Martin, On the scope of the universal-algebraic approach to constraint satisfaction, Logical Methods in Computer Science 8 (3) (2012).

[24] F. Börner, A. A. Bulatov, H. Chen, P. Jeavons, A. A. Krokhin, The complexity of constraint satisfaction games and QCSP, Inf. Comput. 207 (9) (2009) 923–944.

[25] L. J. Stockmeyer, The polynomial-time hierarchy, Theoretical Computer Science 3 (1) (1976) 1 – 22.

[26] D. Kozen, Positive first-order logic is NP-complete, IBM Journal of Research and Development 25 (4) (1981) 327–332.

[27] O. Reingold, Undirected connectivity in log-space, J. ACM 55 (4) (2008) 1–24.