Electronic Colloquium on Computational Complexity, Report No. 23 (2007)

The Complexity of Problems for Quantified Constraints*

Michael Bauland[1], Elmar Böhler[2], Nadia Creignou[3], Steffen Reith[4], Henning Schnoor[1], and Heribert Vollmer[1]

[1] Theoretische Informatik, Universität Hannover, Appelstr. 4, 30167 Hannover, Germany.
`bauland|henning.schnoor|vollmer@thi.uni-hannover.de`
[2] Elektrobit Automotive Software, Frauhenweiherstr. 14, 91058 Erlangen, Germany.
`elmar.boehler@Elektrobit.com`
[3] LIF (UMR 6166), Univ. Méditerranée, Marseille, France.
`creignou@lif.univ-mrs.fr`
[4] University of Applied Sciences, Kurt-Schuhmacher-Ring 18, 65198 Wiesbaden, Germany.
`reith@informatik.fh-wiesbaden.de`

Abstract. In this paper we will look at restricted versions of the evaluation problem, the model checking problem, the equivalence problem, and the counting problem for quantified propositional formulas, both with and without bound on the number of quantifier alternations. The restrictions are such that we consider formulas in conjunctive normal-form with restricted types of clauses (e.g., positive, Horn, linear, etc.). For each of these algorithmic goals we will obtain full complexity classifications, exhibiting on the one hand severe syntactic restrictions of the original problems that are still computationally hard, and on the other hand non-trivial subcases that admit efficient solution algorithms.
Generalizing these results to non Boolean domains, we obtain a number of hardnes results for quantified constraints over arbitrary finite universes.

1 Introduction

Different types of satisfiability problems are important in computational complexity theory because most often they constitute the notorious standard complete problems for many important complexity classes. Very well known is of course the problem SAT, the satisfiability problem for propositional formulas, which, by Cook's Theorem [Coo71] (cf. also [Lev73]), is the first NP-complete problem. In this paper we are interested in quantified propositional formulas. The problem QSAT, the problem to decide if a fully quantified propositional formula is true, is PSPACE-complete [SM73]. Restricting the formulas to be in prenex normal-form with a Σ_k quantifier prefix, i.e., with $k-1$ quantifier alternations starting with an existantial quantifier, we obtain the problem QSAT_k, which is complete for the class Σ_kP of the polynomial hierarchy, also known as Meyer-Stockmeyer hierarchy [MS72].

Besides the *evaluation problem* QSAT or QSAT_k, further important computational goals for quantified propositional logic, following Kleine-Büning and Lettmann [KL99], are the problems of satisfiability, model-checking, and equivalence. For the satisfiability problem, we are given a quantified propositional formula that possibly has some free variables, and we ask if there we can assign Boolean values to the free variables such that then the formula evaluates to true. The complexity of this problem coincides of course with the evaluation problem for problems with an additional first block of existential quantifiers. For the *model checking problem* we are given a not necessarily closed quantified formula and an assignment to the free variables, and we have to decide if this assignment is satisfying. It is not too hard to see that for QSAT_k-formulas this

* Supported in part by the following grants: DFG Vo 630/5-1, 630/5-2, ÉGIDE 05835SH, DAAD D/0205776. Some of the results reported in Sect. 5.2 of this paper already appeared in ECCC Report 05-024.

ISSN 1433-8092

problem is again Σ_kP-complete and if we drop the bound on the number of quantifier alternations the problem is PSPACE-complete. For the *equivalance problem* we are given two not necessarily closed formulas with the same set of free variables, and we have to decide if they have the same set of satisfying assignments. This problem is PSPACE-complete for general quantified formulas and Π_{k+1}P-complete for QSAT_k-formulas [KL99].

Completeness results are a form of lower bounds, and a question arising here is how much we can restrict our problems and still remain complete for important complexity classes. As an example, completeness of QSAT for PSPACE still holds if we restrict the formulas to 3-CNF (formulas in conjunctive normal form with at most 3 literals per clause) [SM73]. The problem QSAT_k remains complete for Σ_kP if we restrict the formulas to 3-CNF for odd values of k, and to 3-DNF (formulas in disjunctive normal form with at most 3 literals per conjunct) for even k, as shown by Wrathall [Wra77].

The more general approach we follow in this paper is the following: We allow arbitrary restrictions (or generalizations) of clauses in the following sense: A constraint relation is just a Boolean relation R of some arity k, i.e., $R \subseteq \{0,1\}^k$. A *constraint* (or *constraint application*) is a formula of the form $R(x_1, \dots, x_k)$ where the x_i are propositional variables. Let S be a finite set of constraint relations. Then an S-formula is a conjunction of constraints where all occuring constraint relations are from S. Such formulas are also called a *Boolean constraint satisfaction problems (CSP)*; S is called *constraint language.* This approach was introduced by Thomas Schaefer [Sch78] who considered the infinite family of problem $\mathsf{CSP}(S)$, the satisfiability problems for S-formulas. Schaefer showed that every such problem is either NP-complete for polynomial-time solvable (i.e., the family avoids the under the assumption P $\neq$ NP infinitely many complexity degrees in between) and moreover obtained an easy criterion to tell, given S, to which case it belongs.

In this paper we study quantified S-formulas/quantified constraint satisfaction problems. For each finite set S of constraint relations, we will determine the complexity of the evaluation problem, the model checking problem, and the equivalence problem for quantified S-formulas. Moreover, as a variant of equivalence, we will also consider the *counting problem*, where we do not ask if two quantified formulas have the same sets of satisfying assignments but want to determine the number of satisfying assignments of a formula. For general formulas this problem is complete for FPSPACE(poly), the class of polynomially length-bounded functions computable in polynomial space [Lad89], and for QSAT_k-formulas it is hard for $\#\cdot\Sigma_k$P-complete under parsimonious reductions (we give an introduction to counting problems and the relevant complexity classes and reducibility notions in Sect. 5.1). For all four computational goals (evaluation, model checking, equivalence, and counting), we obtain full complexity classifications, allowing us, given a set of allowed constraint relations, to determine exactly the complexity of the problem under consideration.

In this way we contribute problems complete for different levels of the polynomial hierarchy that are of a combinatorial structure as simple as possible. The reasoning behind our work is the question how low can we go, i.e., how far we can restrict our formulas, without loosing completeness. What makes a satisfiability problem hard? We will reveal cases for which evaluation, model checking, equivalence, and counting become tractable. For the other cases we provide insight into the sources of hardness by explicitly stating the properties of relations that lead to hard problems.

More precisely we will see that if the set of constraints is Schaefer (that is: all relations are definable by Horn formulas, by dual Horn formulas, by 2CNF formula – the so called bijunctive constraint languages, or by systems of equations over GF[2] – the so called affine constraint languages) then evaluation and model checking are tractable, and in all other cases the general hardness results described above already apply. For equivalence, we obtain tractable cases only for a subset of these constraints: we have to require that all of them are affine, bijunctive or so called *implicative hitting set bounded* (a restriction of Horn formulas) to be able to obtain efficient algorithms. If we are not in one of these cases but Schaefer then equivalence is coNP-complete; in all other cases the general hardness from the above for equivalence holds. For counting, the situation is even worse: only affine relations allow efficient algorithms; if we are not affine but Schaefer we obtain #P-completeness; in all other cases the general hardness (for FPSPACE(poly) or a class $\#{\cdot}\Sigma_k$P, resp.) holds.

Let us illustrate our results with some concrete examples. The relation $\mathrm{NAE} = \{0,1\}^3 \setminus \{(0,0,0),(1,1,1)\}$ (not all components in the 3-tuple are equal, this relation can be shown to be non-Schaefer) leads by Schaefer's Theorem to an NP-complete satisfiability problem. Also, QSAT_k is Σ_kP-complete, the corresponding equivalence problem is Π_{k+1}P-complete, and the counting problem is $\#{\cdot}\Sigma_k$P-complete. For the relation $\mathrm{DUP} = \{0,1\}^3 \setminus \{(0,1,0),(1,0,1)\}$ (all 3-tuples with two consecutive coordinates that share the same value, again a non-Schaefer relation), observe that satisfiability is trivial, since all formulas build using only this relation are satisfied by the constant-0 and the constant-1 truth assignment. However, as for NAE, QSAT_k is Σ_kP-complete, equivalence is Π_{k+1}P-complete, and counting is $\#{\cdot}\Sigma_k$P-complete. For the relation $(x \wedge y) \rightarrow z$ (which is Horn), satisfiability and evaluation of quantified formulas is tractable, but equivalence is coNP-complete for an arbitrary number of quantifier alternations. This is an instance of one of our general results showing that the coNP-completeness of the equivalence problem for the class QHORN of quantified Horn formulas stated in [KL99] holds even for the stricter class of Horn formulas that are at the same time 0-valid and 1-valid. Counting in this case is $\#{\cdot}\Sigma_k$P-complete if the number of quantifier alternations is bounded by $k-1$. A more general summary of our results is given in Fig. 3 on page 24.

As argued in [CKS01], the study of the computational complexity of different algorithmic goals for Boolean constraint satisfaction problems provides a "microcosm of computational complexity theory". The study of this particular family of problems allows a "bird's eye view" of complexity theory and the classes it has created. Our paper adds further support to this thesis. One particular point that will become clear in the course of the paper and that we will address again in the conclusion is the question what type of reductions to use for the study of counting problems. Turing reductions, used by Valiant in his celebrated result that the permanent is #P-complete, as well as the later considered stricter counting reductions turn out to be too coarse, since they cannot distinguish between the levels of the hierarchy of classes $\#{\cdot}\Sigma_k$P. On the other hand, parsimonious reductions are too strict and arguably not suited for the study of counting problems for restricted classes of propositional formulas. In this paper we advocate that complementive reductions (introduced in [BCC+05]) are suitable.

Constraint satisfaction problems have also been studied over larger (non-Boolean) finite or infinite universes. For an overview of recent research in this very active field, the reader is asked to consult [CKV07]. In this paper we will also consider domains of arbitrary finite cardinality. We will show that essentially all hardness results we obtain hold in the more general setting; more precisely the lower bounds in the Boolean case all hold in the general case for all sets of constraint S that have only so called essentially unary or constant polymorphisms. Together with the many results from the literature [BBJK03,Che04a,Che04b,CD05] (cf. also the recent survey [Che06]) that mainly obtain upper bounds for quantified constraints by constructing clever algorithms, this yields not a full complexity classification of the studied problems for general constraints but already quite a detailed picture.

Organization of the paper. In the next section we introduce the reader to the field of constraint satisfaction problems. Important for our results is a Galois connection between the lattice of sets of constraint relations and the lattice of sets of Boolean functions. We provide all necessary background from universal algebra also here in Sect. 2. In Sect. 3 we introduce quantified S-formulas and observe that the Galois connection for not quantified CSPs still helps. In Sects. 4 and 5 we study the complexity of problems for Boolean quantified constraints. We first turn to decision problems (evaluation, model checking, equivalence) for quantified S-formulas in Sects. 4.1

and 4.2. The classification of the evaluation problem has already been obtained in [Hem04], and we give a very short and simple new proof here making use of the Galois connection and Post's lattice. (The original proof from [Hem04] does not make use of universal algebra.) This will turn out important, because the line of argumentation for the evaluation problem—we will show that we do not loose hardness of the algorithmic problem when we simplify the input formulas from general first to 3-CNF, then to the type of relations that is hard in Schaefer's dichotomy, and finally to anything non-Schaefer—will turn out very helpful also for the other problems we study. In Sect. 5 we turn to counting problems. After a detailed introduction of the complexity classes and more importantly the reductions relevant here (Sect. 5.1) we obtain our classification for the Boolean domain in Sect. 5.2. Finally, in Sect. 6 we turn to non-Boolean domains and prove a number of lower bounds. We conclude our paper in Sect. 7 with a short summary and some remaining open questions.

2 Constraint Satisfaction Problems and Closure Properties

Throughout the paper we use the standard correspondence between predicates and relations. We use the same symbol for a predicate and its corresponding relation, the meaning will always be clear from the context. We say that the predicate *represents* the relation. The set $\mathcal{D}$ will denote a finite domain of cardinality $m \geq 2$, $\mathcal{D} = \{0, \ldots, m-1\}$. An n-ary logical relation R is a relation of arity n defined over $\mathcal{D}$, i.e., a set $R \subseteq \mathcal{D}^n$. Let V be a set of variables. A *constraint* is an application of R to an n-tuple of variables from V, i.e., $R(x_1, \ldots, x_n)$. An assignment of values to the variables $I\colon V \to \mathcal{D}$ satisfies the constraint $R(x_1, \ldots, x_n)$ if $(I(x_1), \ldots, I(x_n)) \in R$ holds.

Example 2.1.
- Equivalence, $=^{\mathcal{D}}$, is the binary relation defined by $\{(0,0), \ldots, (m-1, m-1)\}$. Similarly the inequality, $\neq^{\mathcal{D}}$, is defined by $\mathcal{D}^2 \setminus =^{\mathcal{D}}$.
- Given the ternary relation $\mathrm{NAE}^{\mathcal{D}} = \mathcal{D}^3 \setminus \{(0,0,0), \ldots, (m-1, m-1, m-1)\}$, the constraint $\mathrm{NAE}^{\mathcal{D}}(x_1, x_2, x_3)$ is satisfied if and only if not all variables are assigned the same value. We write NAE^m for $\mathrm{NAE}^{\mathcal{D}}$ with $|\mathcal{D}| = m$.
- The Boolean constraint $\mathrm{R}_{n/m}(x_1, \ldots, x_m)$ is satisfied if exactly n of the m variables are assigned to 1.

Let S be a non-empty finite set of non-empty relations defined over $\mathcal{D}$; such sets are often called *constraint languages.* An *S-formula* is a finite conjunction of *S-clauses*, $\varphi = c_1 \wedge \cdots \wedge c_k$, where each S-clause c_i is a constraint application of some logical relation $R \in S$. An assignment I satisfies φ if it satisfies all clauses c_i. We denote by $\mathsf{sat}(\varphi)$ the set of satisfying assignments of a formula φ. We denote by $\mathsf{CSP}(S)$ the satisfiability problem for S-formulas. For a relation R, we often write $\mathsf{CSP}(R)$ instead of $\mathsf{CSP}(\{R\})$. The acronym "CSP" stands for *constraint satisfaction problem.*

We now show with two examples how CSPs can be used to express important computational problems in computer science:

Example 2.2.
- The well-known 3-SAT problem can be seen as the constraint satisfaction problem over the set $\mathrm{S}_{\mathsf{3SAT}} = \{(x_1 \vee x_2 \vee x_3), (\overline{x_1} \vee x_2 \vee x_3), (\overline{x_1} \vee \overline{x_2} \vee x_3), (\overline{x_1} \vee \overline{x_2} \vee \overline{x_3})\}$.
- The 3-Colorability problem can be seen as the constraint satisfaction problem using only the inequality relation over the three-element domain.

Given a set S of relations, in order to study the complexity of $\mathsf{CSP}(S)$ we will be interested in the expressive power of S, which can be measured by the set $\mathsf{COQ}(S)$ of all relations that can be represented by formulas of the form

$$\varphi(x_1, \ldots, x_k) \quad = \quad \exists y_1 \exists y_2 \cdots \exists y_l \; \varphi(x_1, \ldots, x_k, y_1, \ldots, y_l),$$

where φ is an S-formula. Such formulas are also called *conjunctive-queries.*

Throughout the text we refer to different types of Boolean constraint relations following Schaefer's terminology [Sch78]. We say that a Boolean relation R is *1-valid* if $(1,\ldots,1) \in R$, *0-valid* if $(0,\ldots,0) \in R$, *Horn* (*dual Horn*, resp.) if R can be represented by a conjunctive normal form (CNF) formula having at most one unnegated (negated, resp.) variable in each clause, *bijunctive* if it can be represented by a CNF formula having at most two variables in each clause, *affine* if it can be represented by a conjunction of linear functions, i.e., a CNF formula with $\oplus$-clauses (XOR-CNF), *complementive* if for each $(\alpha_1,\ldots,\alpha_n) \in R$, also $(\neg\alpha_1,\ldots,\neg\alpha_n) \in R$.

A set S of Boolean relations is called 0-valid (1-valid, Horn, dual Horn, affine, bijunctive, complementive) if every relation in S has this property. Finally a set S of Boolean relations is called *Schaefer* if it is Horn, dual Horn, affine, or bijunctive.

Given a Boolean relation R the following well-known closure properties determine the structure of R (operations are applied coordinate-wise on vectors, maj is the ternary majority function, which yields 1 if and only if at least two of its arguments are 1) [Sch78,CKS01].

- R is Horn if and only if $m, m' \in R$ implies $m \wedge m' \in R$.
- R is dual Horn if and only if $m, m' \in R$ implies $m \vee m' \in R$.
- R is bijunctive if and only if $m, m', m'' \in R$ implies $\mathrm{maj}(m, m', m'') \in R$.
- R is affine if and only if $m, m', m'' \in R$ implies $m \oplus m' \oplus m'' \in R$.
- R is complementive if and only if $m \in R$ implies $\neg m \in R$.

The notion of closure properties of a relation has been defined more generally, see for instance [JCG97,Pip97]. Let $f\colon \mathcal{D}^k \to \mathcal{D}$ be a k-ary function. We say that R is *closed under* f, or that f is a *polymorphism* of R, if for any choice of k vectors $m_1,\ldots,m_k \in R$, not necessarily distinct, we have that

$$\Big(f\big(m_1[1],\ldots,m_k[1]\big),\ f\big(m_1[2],\ldots,m_k[2]\big),\ \ldots,\ f\big(m_1[n],\ldots,m_k[n]\big)\Big) \in R,$$

i.e., the vector constructed coordinate-wise from m_1, ..., m_k by means of f belongs to R.

We denote by $\mathrm{Pol}(R)$ the set of all polymorphisms of R and by $\mathrm{Pol}(S)$ the set of functions that are polymorphisms of every relation in S. It turns out that $\mathrm{Pol}(S)$ is a *clone* for every set of relations S, i.e., $\mathrm{Pol}(S)$ contains all projection functions and is closed under superposition (composition of functions), see, e.g., [Pip97].

A Galois correspondence exists between the sets of functions $\mathrm{Pol}(S)$ and the sets of relations S. An introduction to this correspondence can be found in [Pip97,Pös01] and a comprehensive study in [PK79,Lau06]. This theory helps us to get elegant and short proofs for complexity results concerning constraint satisfaction problems, see, e.g., [JCG97,Dal00,BCRV04]. Indeed, it shows that the smaller the set of polymorphisms is, the more expressive the corresponding conjunctive queries are, which is the cornerstone for applying the algebraic method to complexity. The following proposition can be found, e.g., in [Dal00].

Proposition 2.3. *Let S_1 and S_2 be constraint languages defined over $\mathcal{D}$. If the inclusion $\mathrm{Pol}(S_2) \subseteq \mathrm{Pol}(S_1)$ holds, then $\mathsf{COQ}(S_1) \subseteq \mathsf{COQ}(S_2 \cup \{=^{\mathcal{D}}\})$.*

This result was used in [JCG97] to obtain the following complexity result.

Theorem 2.4. *Let S_1 and S_2 be constraint languages. If the inclusion $\mathrm{Pol}(S_2) \subseteq \mathrm{Pol}(S_1)$ holds, then $\mathsf{CSP}(S_1)$ is polynomial-time reducible to $\mathsf{CSP}(S_2)$.*

A number of results on the complexity of CSP have been obtained via this approach (see, e.g., [JCG97,Bul06]). In particular, the well-known Schaefer's dichotomy theorem can be proved in this way by using *Post's lattice* (see, e.g., [BCRV04]).

Theorem 2.5. [Sch78] *Let S be a Boolean constraint language. If S is Schaefer, or* 0- *or* 1-*valid, then* $\mathsf{CSP}(S)$ *is in* P*, otherwise* $\mathsf{CSP}(S)$ *is* NP-*complete.*

To obtain a full classification for the just given and many more results, one starts with a constraint language S and then looks at the set of its polymorphisms $\mathrm{Pol}(S)$. Since this is a clone, one then enters a case distinction for all possible clones. In the case of the Boolean universe, all clones and all inclusions among them are known, see Fig. 1. This figure is nowadays known as *Post's lattice* and is described, e.g., in [Pip97,BCRV03]. For the purpose of this paper, we define the clones by simply giving a basis for each of them, see Fig. 2, i.e., the third column of the table gives for each clone its defining basis. Here, if B is such a basis, then the corresponding clone, denoted by $[B]$, is the smallest set of Boolean functions that contains all functions from B and all projections (i.e., all functions $\mathrm{I}^n_k(x_1, \dots, x_n) = x_k$) and is closed under composition of functions. One function appearing in the bases that is maybe not so familiar is the threshold function T^n_k, where $\mathrm{T}^n_k(x_1, \dots, x_n) = 1 \iff \sum_{i=1}^n x_i \geq k$. The previously mentioned function maj is just T^3_2.

Further properties of the lattice will be introduced in the development of this paper as needed. We only mention one further concept already here, that of *duality*.

We say a function f is the dual function of g if they both have the same arity n and for all $a_1, \dots, a_n \in \{0, 1\}$ we have $f(a_1, \dots, a_n) = \overline{g(\overline{a_1}, \dots, \overline{a_n})}$. We define $\mathrm{dual}(f)$ to be the dual function of f and for a set of Boolean functions B we let $\mathrm{dual}(B) = \{\, \mathrm{dual}(f) \mid f \in B \,\}$. A function is self-dual if it coincides with its dual.

For each clone B, the set $\mathrm{dual}(B)$ is again a clone. Looking at Fig. 1, imagine a symmetry axis through BF and I_2. Now, for each class on the one side of the axis the dual one is the mirror image on the other side of the axis. For classes B located *on* the axis, we have $\mathrm{dual}(B) = B$.

Looking at the list of bases for Post's lasses, the above characterizations of the Schaefer and other classes of formulas in terms of Boolean operations ($\wedge$, $\vee$, maj, $\oplus$) can now be stated as follows:

$\mathrm{Pol}(R) \supseteq \mathrm{E}_2$	$\Leftrightarrow$	R is Horn	$\mathrm{Pol}(R) \supseteq \mathrm{D}_2$	$\Leftrightarrow$	R is bijunctive
$\mathrm{Pol}(R) \supseteq \mathrm{V}_2$	$\Leftrightarrow$	R is dual Horn	$\mathrm{Pol}(R) \supseteq \mathrm{L}_2$	$\Leftrightarrow$	R is affine
$\mathrm{Pol}(R) \supseteq \mathrm{I}_0$	$\Leftrightarrow$	R is 0-valid	$\mathrm{Pol}(R) \supseteq \mathrm{N}$	$\Leftrightarrow$	R is complementive, 0- and 1-valid
$\mathrm{Pol}(R) \supseteq \mathrm{I}_1$	$\Leftrightarrow$	R is 1-valid	$\mathrm{Pol}(R) \supseteq \mathrm{N}_2$	$\Leftrightarrow$	R is complementive
$\mathrm{Pol}(R) \supseteq \mathrm{I}_2$	$\Leftrightarrow$	R is any relation			

Looking at the inclusion structure of the lattice, we note in particular that this means that a relation R is not Schaefer if and only if $\mathrm{Pol}(R) \subseteq \mathrm{N}$. One special case here is that of relations R for which $\mathrm{Pol}(R) = \mathrm{N}$, this is the class of relations that are not Schaefer but complementive, 0-valid, and 1-valid. As for the Schaefer cases, for these the satisfiability problem is still polynomial-time decidable (in fact, all formulas here are satisfiable, since the all-1 or all-0 assignment will always be satisfying). Only if we drop the requirements of R being 0-valid or 1-valid, we arrive at the hard cases for satisfiability. This is the set of relations R for which $\mathrm{Pol}(R) \subseteq \mathrm{N}_2$. Hence, a reformulation of Schaefer's Theorem 2.5 is the following:

Corollary 2.6. *Let S be a constraint language. If* $\mathrm{Pol}(S) \subseteq \mathrm{N}_2$ *then* $\mathsf{CSP}(S)$ *is* NP-*complete, in all other cases* $\mathsf{CSP}(S) \in \mathrm{P}$.

We need to introduce another class of formulas, namely the class of IHSB (for implicative hitting set bounded) formulas. These formulas form a subclass of and are less expressive than the class of Horn and dual Horn formulas; for more background the reader is asked to consult [CKS01].

A clause is said to be IHSB$-$ if it is of one of the following types: (x_i), $(\neg x_{i_1} \vee x_{i_2})$ or $(\neg x_{i_1} \vee \dots \vee \neg x_{i_k})$ for some $k \geq 1$. Dually, a clause is said to be IHSB+ if it is of one of the following types: $(\neg x_i)$, $(\neg x_{i_1} \vee x_{i_2})$ or $(x_{i_1} \vee \dots \vee x_{i_k})$ for some $k \geq 1$. Finally, a formula is said to be IHSB$-$ (resp. IHSB+) if all its clauses are IHSB$-$ (resp. IHSB+).

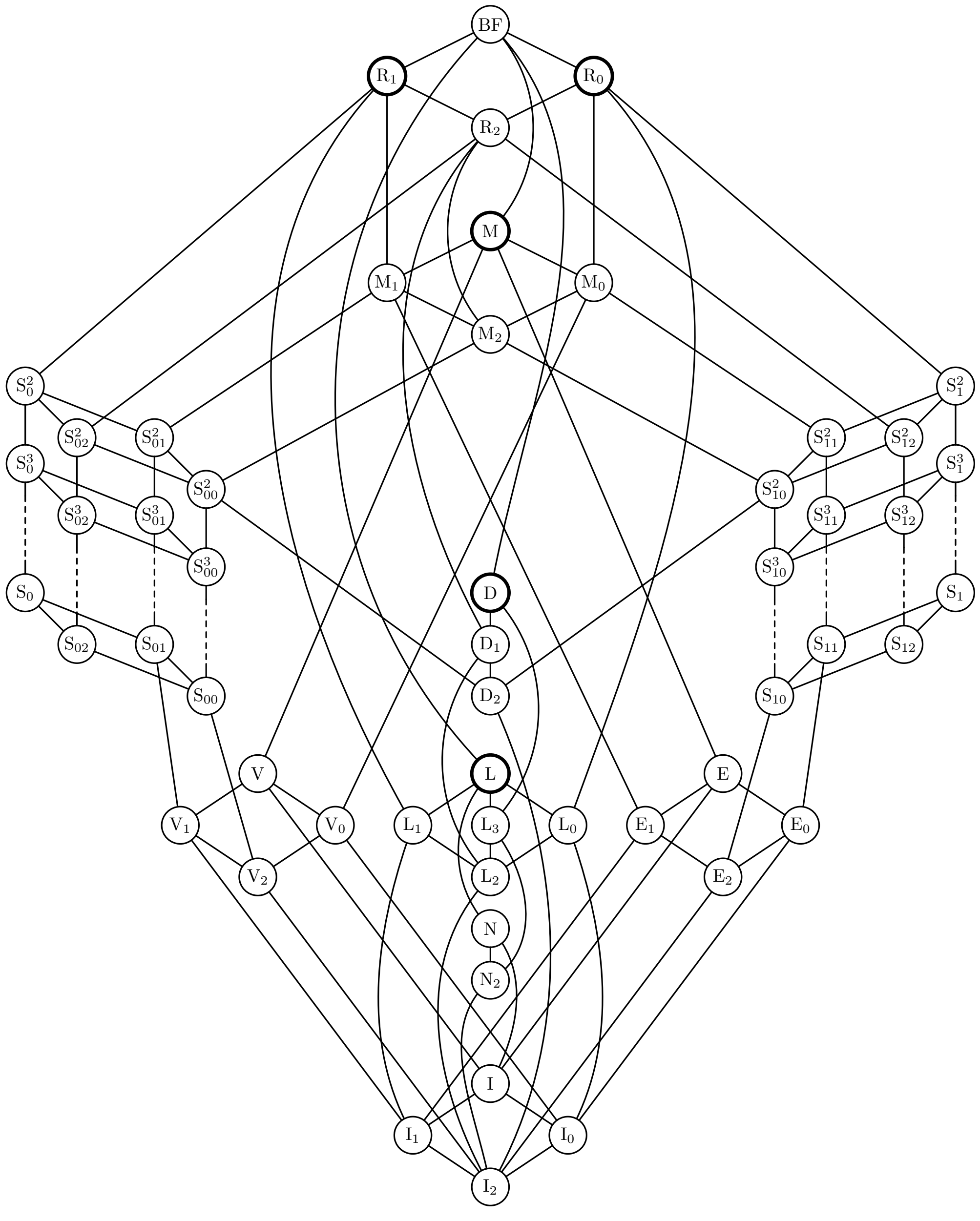

Fig. 1. Graph of all Boolean clones

Class	Description	Base
BF	all Boolean functions	$\{\land, \neg\}$
R_0	0-reproducing functions	$\{\land, \oplus\}$
R_1	1-reproducing functions	$\{\lor, x \oplus y \oplus 1\}$
R_2	$R_1 \cap R_0$	$\{\lor, x \land (y \oplus z \oplus 1)\}$
M	monotone functions	$\{\land, \lor, 0, 1\}$
M_1	$M \cap R_1$	$\{\land, \lor, 1\}$
M_0	$M \cap R_0$	$\{\land, \lor, 0\}$
M_2	$M \cap R_2$	$\{\land, \lor\}$
S_0^n	functions that are 0-separating of degree n	$\{\rightarrow, \mathrm{dual}(T_n^{n+1})\}$
S_0	0-separating functions	$\{\rightarrow\}$
S_1^n	functions that are 1-separating of degree n	$\{x \land \overline{y},\ T_n^{n+1}\}$
S_1	1-separating functions	$\{x \land \overline{y}\}$
S_{02}^n	$S_0^n \cap R_2$	$\{x \lor (y \land \overline{z}), \mathrm{dual}(T_n^{n+1})\}$
S_{02}	$S_0 \cap R_2$	$\{x \lor (y \land \overline{z})\}$
S_{01}^n	$S_0^n \cap M$	$\{\mathrm{dual}(T_n^{n+1}), 1\}$
S_{01}	$S_0 \cap M$	$\{x \lor (y \land z), 1\}$
S_{00}^n	$S_0^n \cap R_2 \cap M$	$\{x \lor (y \land z), \mathrm{dual}(T_n^{n+1})\}$
S_{00}	$S_0 \cap R_2 \cap M$	$\{x \lor (y \land z)\}$
S_{12}^n	$S_1^n \cap R_2$	$\{x \land (y \lor \overline{z}), T_n^{n+1}\}$
S_{12}	$S_1 \cap R_2$	$\{x \land (y \lor \overline{z})\}$
S_{11}^n	$S_1^n \cap M$	$\{T_n^{n+1}, 0\}$
S_{11}	$S_1 \cap M$	$\{x \land (y \lor z), 0\}$
S_{10}^n	$S_1^n \cap R_2 \cap M$	$\{x \land (y \lor z), T_n^{n+1}\}$
S_{10}	$S_1 \cap R_2 \cap M$	$\{x \land (y \lor z)\}$
D	self-dual functions	$\{x\overline{y} \lor x\overline{z} \lor \overline{yz}\}$
D_1	$D \cap R_2$	$\{xy \lor x\overline{z} \lor y\overline{z}\}$
D_2	$D \cap M$	$\{xy \lor yz \lor xz\}$
L	linear functions	$\{\oplus, 1\}$
L_0	$L \cap R_0$	$\{\oplus\}$
L_1	$L \cap R_1$	$\{\equiv\}$
L_2	$L \cap R_2$	$\{x \oplus y \oplus z\}$
L_3	$L \cap D$	$\{x \oplus y \oplus z \oplus 1\}$
V	$\lor$-functions plus constant functions	$\{\lor, 0, 1\}$
V_0	$[\{\lor\}] \cup [\{0\}]$	$\{\lor, 0\}$
V_1	$[\{\lor\}] \cup [\{1\}]$	$\{\lor, 1\}$
V_2	$[\{\lor\}]$	$\{\lor\}$
E	$\land$-functions plus constant functions	$\{\land, 0, 1\}$
E_0	$[\{\land\}] \cup [\{0\}]$	$\{\land, 0\}$
E_1	$[\{\land\}] \cup [\{1\}]$	$\{\land, 1\}$
E_2	$[\{\land\}]$	$\{\land\}$
N	$[\{\neg\}] \cup [\{0\}] \cup [\{1\}]$	$\{\neg, 1\}$, $\{\neg, 0\}$
N_2	$[\{\neg\}]$	$\{\neg\}$
I	$I_2 \cup [\{1\}] \cup [\{0\}]$	$\{0, 1\}$
I_0	$I_2 \cup [\{0\}]$	$\{0\}$
I_1	$I_2 \cup [\{1\}]$	$\{1\}$
I_2	all projections	$\emptyset$

Fig. 2. List of all Boolean clones with their defining bases

As usual a Boolean relation R is said to be IHSB$-$ (resp. IHSB$+$) if R can be represented by a CNF formula which is IHSB$-$ (resp. IHSB$+$). Finally, a constraint language S is said to be IHSB$-$ (resp. IHSB$+$) if every relation in S is IHSB$-$ (resp. IHSB$+$).

As for the above introduced classes, IHSB relations can be characterized by their polymorphism, as follows immediately from [BRSV05]:

$$\begin{aligned}\mathrm{Pol}(R) \supseteq \mathrm{S}_{10} &\Leftrightarrow R \text{ is IHSB}- \\ \mathrm{Pol}(R) \supseteq \mathrm{S}_{00} &\Leftrightarrow R \text{ is IHSB}+\end{aligned}$$

3 Quantified Problems

In this paper we consider the more general framework of quantified constraint satisfaction problems, which are defined as follows.

Let S be a finite set of relations defined over the domain $\mathcal{D}$. An instance of $\mathsf{QCSP}(S)$ is a closed formula of the form $Q_1x_1Q_2x_2\ldots Q_nx_n\phi$, where $Q_1,\ldots,Q_n$ are arbitrary quantifiers and ϕ is an S-formula. The question is whether the sentence is true, i.e., if there exists, for every assignment Π to the universally quantified variables, an assignment E to the existentially quantified variables, where the value for each existentially quantified variable only depends on the values of universal variables quantified before, such that the formula ϕ holds when assigning these values to the variables. One can use an exhaustive search algorithm to show that $\mathsf{QCSP}(S)$ is always in PSPACE.

The problem QSAT of deciding, whether a given closed quantified Boolean formula is true, is PSPACE-complete [SM73]. This problem remains PSPACE-complete if we restrict the formulas to 3-CNF [Sto77]. It is worth noticing that the Boolean case still displays a dichotomy for satisfiability of quantified S-formulas. The following theorem was stated by Schaefer only for constraint languages which include the constants, but his proof ideas are sufficient to show the complete classification.

Theorem 3.1. [Sch78,Dal97,CKS01] *Let S be aBoolean constraint language. If S is Schaefer, then* $\mathsf{QCSP}(S)$ *is in* P*, otherwise* $\mathsf{QCSP}(S)$ *is* PSPACE*-complete.*

In this paper, we are interested in quantified constraint satisfaction problems in which the number of quantifier alternations is bounded. These problems are prototypical for the *polynomial-time hierarchy* (PH for short), which was defined by Meyer and Stockmeyer [MS72]. Following the notation of [Pap94], $\Sigma_0\mathrm{P} = \Pi_0\mathrm{P} = \mathrm{P}$ and for all $i \geq 0$, $\Sigma_{i+1}\mathrm{P} = \mathrm{NP}^{\Sigma_i\mathrm{P}}$ and $\Pi_{i+1}\mathrm{P} = \mathrm{coNP}^{\Sigma_i\mathrm{P}}$. The set QSAT_k is the set of all closed, true quantified Boolean formulas with $k-1$ quantifier alternations, starting with an $\exists$-quantifier. For all $k \geq 1$, QSAT_k is complete for $\Sigma_k\mathrm{P}$. This problem remains $\Sigma_k\mathrm{P}$-complete if we restrict the Boolean formula to be 3-CNF for k odd, and 3-DNF for k even [Wra77]. Note that for formulas starting with a universal quantifier, the satisfiability problem for formulas in disjunctive normal form is complete for the levels of the hierarchy. Since disjunctive normal forms cannot be naturally modeled in a constraint satisfaction context, in order to generalize QSAT_k to arbitrary set of constraints S and to get complete problems for the levels of the polynomial hierarchy, we consider the unsatisfiability problem for these cases. Thus, we adopt the following definition for $\mathsf{QCSP}_k(S)$ from [Hem04]:

Let S be set a of relations over the domain $\mathcal{D}$ and let $k \geq 1$.

- A $\Sigma_k(S)$-formula is a formula of the form $\varphi = \exists X_1\forall X_2\ldots Q_kX_k\psi$,
- a $\Pi_k(S)$-formula is a formula of the form $\varphi = \forall X_1\exists X_2\ldots \overline{Q}_kX_k\psi$,

where the X_j, $j = 1,\ldots,k$, are disjoint sets of variables, $Q_i = \exists$ and $\overline{Q}_i = \forall$ if i is odd, $Q_i = \forall$ and $\overline{Q}_i = \exists$ if i is even, and ψ is a quantifier-free S-formula defined on $\bigcup_j X_j \cup Z$ for some set Z of *free variables*, and is called the *matrix* of Φ. For k odd, a $\mathsf{QCSP}_k(S)$-formula is a $\Sigma_k(S)$-formula, and for k even, a $\mathsf{QCSP}_k(S)$-formula is a $\Pi_k(S)$-formula.

Observe that for all k, the innermost quantifier of a $\mathsf{QCSP}_k(S)$-formula is existential. In the following, making use of the above definition of the quantifiers Q_i, we will denote such a formula by $\varphi = Q_k X_1 Q_{k-1} X_2 \ldots \exists X_k \psi$.

Problem:	$\mathsf{QCSP}_k(S)$
Input:	a closed $\mathrm{QCSP}_k(S)$-formula ϕ
Question:	If k is odd: Is ϕ true?
	If k is even: Is ϕ false?

Note that $\mathsf{QCSP}_k(S)$ belongs to $\Sigma_k\mathrm{P}$ for each $k \geq 1$. Moreover, according to Wrathall's result [Wra77], $\mathsf{QCSP}_k(\mathrm{S}_{\mathsf{3SAT}})$ (see Example 2.2) is $\Sigma_k\mathrm{P}$-complete.

The following proposition states that the Galois connection between sets of relations and their closure properties still applies to quantified problems with bounded alternations.

Proposition 3.2. *Let S_1 and S_2 be constraint languages over the same domain, and let $k \geq 1$. If the inclusion $\mathrm{Pol}(S_2) \subseteq \mathrm{Pol}(S_1)$ holds, then $\mathsf{QCSP}_k(S_1)$ is logspace many-one reducible to $\mathsf{QCSP}_k(S_2)$.*

Proof. If $\mathrm{Pol}(S_2) \subseteq \mathrm{Pol}(S_1)$, then due to Proposition 2.3 one can express every relation from S_1 with an existential $S_2 \cup \{=^{\mathcal{D}}\}$-formula. We locally replace every S_1-constraint by its equivalent $S_2 \cup \{=^{\mathcal{D}}\}$-formula and move the additional existential variables to the right end of the quantifier sequence. Since in every $\mathsf{QCSP}_k(S)$-formula the last quantifier is $\exists$, we end up with a $\mathsf{QCSP}_k(S_2 \cup \{=^{\mathcal{D}}\})$-formula equivalent to the original formula.

We now remove the equality constraints. We check if there are variables x and y such that y is $\forall$-quantified after x is quantified with an $=$-path from x to y. In this case, the formula is false. Otherwise, all $=$-connected components of variables consist of variables of which at most the first one, x, is universally quantified. We can rename all these variables to x and delete the corresponding existential quantifiers. The complexity of this procedure is dominated by undirected graph accessibility, which is in logspace due to [Rei05]. $\square$

Contrary to [BBJK03, Theorem 4], we cannot restrict our attention to surjective polymorphisms, because in the context of bounded quantifier alternation, this does not yield sharp reductions. Börner et al. prove that if S_1, S_2 are constraint languages such that all surjective polymorphisms of S_2 are polymorphisms of S_1 then $\mathsf{QCSP}(S_1)$ reduces to $\mathsf{QCSP}(S_2)$. However, the reductions obtained from this theorem increase the number of quantifier alternations. Thus, while their reduction is a useful tool for the case of an unbounded number of quantifier alternations, it cannot be applied to prove completeness results for individual levels of the polynomial hierarchy.

4 Decision Problems for Quantified Boolean CSPs

In this and the next section we turn to the study of different computational goals for quantified constraints over the Boolean domain, before we study domains of higher cardinality in Sect. 6.

4.1 Evaluation and Model Checking

We turn first to the complexity of deciding if a given quantified formula over the Boolean domain is true, and consider related decision problems later in this section.

As it turns out, the relation NAE is central for our development, and as a starting point we consider formulas with this logical relation only.

Lemma 4.1. *For $k \geq 1$, $\mathsf{QCSP}_k(\mathrm{NAE}^2)$ is $\Sigma_k\mathrm{P}$-complete under logspace reductions.*

Proof. Let I_2 be the clone containing all projections. This is the smallest clone, contained in all other clones. Since $\mathrm{Pol}(\mathrm{R}_{1/3}) = \mathrm{I}_2 \subseteq \mathrm{Pol}(\mathrm{S}_{\mathsf{3SAT}})$, Proposition 3.2 states that $\mathsf{QCSP}_k(\mathrm{S}_{\mathsf{3SAT}})$ is logspace many-one reducible to $\mathsf{QCSP}_k(\mathrm{R}_{1/3})$. Since $\mathsf{QCSP}_k(\mathrm{S}_{\mathsf{3SAT}})$ is complete for $\Sigma_k\mathrm{P}$, it suffices to show $\mathsf{QCSP}_k(\mathrm{R}_{1/3}) \leq_m^{\log} \mathsf{QCSP}_k(\mathrm{NAE}^2)$.

Let φ be a $\mathsf{QCSP}_k(\mathrm{R}_{1/3})$-formula, $\varphi = Q_kX_1Q_{k-1}X_2\ldots\exists X_k \bigwedge_{j=1}^{p} \mathrm{R}_{1/3}(x_{j_1}, x_{j_2}, x_{j_3})$. For each constraint $\mathrm{R}_{1/3}(x_{j_1}, x_{j_2}, x_{j_3})$, introduce the following conjunction of NAE^2 constraints:

$$\mathrm{R}_{2/4}(x_{j_1}, x_{j_2}, x_{j_3}, t) = \bigwedge_{j \neq k \in \{j_1, j_2, j_3\}} \mathrm{NAE}^2(x_j, x_k, t) \wedge \mathrm{NAE}^2(x_{j_1}, x_{j_2}, x_{j_3}).$$

Let $\varphi' = Q_ktQ_kX_1Q_{k-1}X_2\ldots\exists X_k \bigwedge_{j=1}^{p} \mathrm{R}_{2/4}(x_{j_1}, x_{j_2}, x_{j_3}, t)$. Since obviously $\mathrm{R}_{1/3}(x, y, z) = \mathrm{R}_{2/4}(x, y, z, 1)$, the formula $\varphi'[t/1]$ (that is, φ with every occurrence of t replaced by 1) is true iff φ is true. Since $\mathrm{R}_{1/3}(\bar{x}, \bar{y}, \bar{z}) = \mathrm{R}_{2/4}(x, y, z, 0)$, the formula $\varphi'[t/0]$ is true iff $\mathrm{Ren}(\varphi)$ is true, where $\mathrm{Ren}(\varphi)$ is obtained from φ by renaming all variables x by their negation $\bar{x}$. Finally, since $\mathrm{Ren}(\varphi)$ is true iff φ is true, we proved that φ is true if and only if φ' is true. ☐

Since $\mathrm{Pol}(\mathrm{NAE}^2) = \mathrm{N}_2$, we proved that, if $\mathrm{Pol}(S) = \mathrm{N}_2$, then $\mathsf{QCSP}_k(S)$ is complete for $\Sigma_k\mathrm{P}$. We now show how this result generalizes to the case where we also have constant polymorphisms, i.e., the entire clone N, and therefore to the "maximal non-Schaefer" case.

Lemma 4.2. *There exists a Boolean relation R_0 such that $\mathrm{Pol}(R_0) = \mathrm{N}$ and for all $k \geq 2$, $\mathsf{QCSP}_k(\mathrm{NAE}^2)$ reduces to $\mathsf{QCSP}_k(R_0)$ under logspace reductions.*

Proof. We define R_0 to be the relation

$$R_0 = \left\{ (u, v, x_1, x_2, x_3) \,\middle|\, u = v \text{ or } \mathrm{NAE}^2(x_1, x_2, x_3) \right\}.$$

It is easy to see that $\mathrm{Pol}(R_0)$ contains all the constants as well as the negation, thus $\mathrm{Pol}(R_0) = \mathrm{N}$.

Now we prove that $\mathsf{QCSP}_k(\mathrm{NAE}^2)$, which is complete for $\Sigma_k\mathrm{P}$, can be reduced to $\mathsf{QCSP}_k(R_0)$ in logarithmic space. Let

$$\varphi = Q_kX_1\ldots\exists X_k \bigwedge_{j=1}^{p} \mathrm{NAE}^2(x_{j_1}, x_{j_2}, x_{j_3})$$

be an instance of $\mathsf{QCSP}_i(\mathrm{NAE}^2)$. We define

$$\varphi' = Q_kX_1\ldots\forall X_{i-1}\forall u\forall v\exists X_k \bigwedge_{j=1}^{p} R_0(u, v, x_{j_1}, x_{j_2}, x_{j_3}).$$

It is clear that φ is true if and only if φ' is true, concluding the proof of the lemma. ☐

Thus we obtain the following classification of the complexity of $\mathsf{QCSP}_k(S)$ for $k \geq 2$—note that the case $k = 1$ is given in Theorem 2.5:

Theorem 4.3. *Let S be a Boolean constraint language, and let $k \geq 2$. If S is Schaefer, then $\mathsf{QCSP}_k(S)$ is in* P*, otherwise $\mathsf{QCSP}_k(S)$ is $\Sigma_k\mathrm{P}$-complete under logspace reductions.*

Proof. The polynomial cases follow from Theorem 3.1. According to the closure properties of a non-Schaefer set (see [BCRV04], Section 2), the case $\mathrm{Pol}(S) = \mathrm{N}$ remains. Now the theorem follows from Lemma 4.1, Lemma 4.2, and Proposition 3.2. ☐

Theorem 4.3 settles the Boolean case completely, thus reproving via the algebraic approach a result first obtained by E. Hemaspaandra [Hem04]. For the present paper, this theorem is just a first technical step to show how the Galois connection can work in the quantified context. The reason for us to present this re-proof of Hemaspaandra's theorem is that we will obtain results that concern the counting complexity of the quantified constraint satisfaction problem with a very similar technique later in Section 5.

A slight variant of the evaluation problem $\mathsf{QCSP}_k(S)$ is the *model checking problem* (cf. [KL99]), where we are given a not necessarily closed quantified formula and an assignment to its free variables, and we ask if this assignment is satisfying.

We generalize the notation $\phi[x/\alpha]$ from the above to the simultaneous substitution of several variables, i.e., for a formula φ and variables $x_1, \ldots, x_n$, the formula $\varphi[x_1/\alpha_1, \ldots, x_n/\alpha_n]$ is the formula φ with every occurence of x_i replaced by α_i, for $1 \leq i \leq n$.

Problem:	$\mathsf{QMC}_k(S)$
Input:	a $\mathsf{QCSP}_k(S)$-formula $\varphi(z_1, \ldots, z_n)$ with free variables $z_1, \ldots, z_n$ and values $\alpha_1, \ldots, \alpha_n \in \mathcal{D}$
Question:	Does $\varphi[z_1/\alpha_1, \ldots, z_n/\alpha_n]$ hold?

As usual, QMC denotes the version with unbounded number of quantifier alternations.

Though our proofs will actually not depend on it, we note that the Galois connection works as well for model checking:

Proposition 4.4. *Let S_1 and S_2 be constraint languages over the same domain such that* $\mathrm{Pol}(S_2) \subseteq \mathrm{Pol}(S_1)$. *Then* $\mathsf{QMC}(S_1) \leq_m^{\log} \mathsf{QMC}(S_2)$ *and* $\mathsf{QMC}_k(S_1) \leq_m^{\log} \mathsf{QMC}_k(S_2)$ *for all* k.

Proof. In the same way as in the proof for Proposition 3.2, we can transform S_1-formulas into S_2-formulas. When identifying two variables x and y in renaming every occurrence of y to x, and the assignment given in the instance does not give the same value to x and y, we produce a false instance. In this way we ensure that the original assignment is a solution for the original formula if and only if it is one for the new formula. □

Since evaluation for any quantified formula of the appropriate quantifier alternation is in the corresponding class of the polynomial hierarchy, the following is obvious:

Proposition 4.5. *Let S be a constraint language and $k \in \mathbb{N}$. Then* $\mathsf{QMC}_k(S) \in \Sigma_k\mathrm{P}$ *if k is odd, and* $\mathsf{QMC}_k(S) \in \Pi_k\mathrm{P}$ *if k is even. Furthermore,* $\mathsf{QMC}(S) \in \mathrm{PSPACE}$.

It is obvious that the model checking problem is closely related to the evaluation problem for quantified formulas. Since for a fully quantified formula, i.e., a formula with no free variables, model checking is the same as evaluation, we immediately get the following:

Proposition 4.6. *Let S be a constraint language and $k \in \mathbb{N}$.*

1. *If k is odd, then* $\mathsf{QCSP}_k(S) \leq_m^{\log} \mathsf{QMC}_k(S)$.
2. *If k is even, then* $\overline{\mathsf{QCSP}_k(S)} \leq_m^{\log} \mathsf{QMC}_k(S)$.
3. $\mathsf{QCSP}(S) \leq_m^{\log} \mathsf{QMC}(S)$.

This, combined with the results for evaluation from above, leads to the following Corollary—note that the case $k = 1$ is a direct consequence of Schaefer's result, since in [Sch78], he proves that the satisfiability problem with constants is NP-complete for non-Schaefer constraint languages.

Corollary 4.7. *Let S be a Boolean constraint language such that* $\mathrm{Pol}(S) \subseteq \mathrm{N}$. *Then* $\mathsf{QMC}_k(S)$ *is complete for* $\Sigma_k\mathrm{P}$ *if k is odd, and complete for* $\Pi_k\mathrm{P}$ *if k is even.*

Now it is easy to see with a look at Post's lattice that for a constraint language S which is Schaefer, the language $S' = S \cup \{x, \overline{x}\}$ is Schaefer as well. Since the model checking problem can be solved by replacing the variables with constants, and the evaluation problem for quantified S-formulas is in P if S is Schaefer, we get the following:

Proposition 4.8. *Let S be a constraint language that is Schaefer. Then $\mathsf{QMC}(S)$ is solvable in polynomial time.*

Proof. Formally, given $\varphi(z_1, \dots, z_n) = Q_i X_1 Q_{i-1} X_2 \dots \exists X_i \psi(X_1 \cup \dots \cup X_k \cup \{z_1, \dots, z_n\})$ and values $\alpha_1, \dots, \alpha_n$, we add a clause z_i if $\alpha_i = 1$, and a clause $\overline{z_i}$ if $\alpha_i = 0$. We existentially quantify the z_i variables in the last $\exists$-block. This is then an instance of $\mathsf{QCSP}(S')$, which is in P by the above observation. □

Therefore, the complete classification of the complexity of the model checking problem for the Boolean case is as follows.

Theorem 4.9. *Let S be a Boolean constraint language.*

1. *If S is Schaefer, then $\mathsf{QMC}(S)$ is solvable in polynomial time.*
2. *Otherwise, $\mathsf{QMC}(S)$ is complete for PSPACE, and furthermore, $\mathsf{QMC}_k(S)$ is complete for $\Sigma_k\mathrm{P}$ if k is odd, and complete for $\Pi_k\mathrm{P}$ if k is even.*

4.2 The Equivalence Problem

We now turn to the considerably more complicated setting where we want to check equivalence of two quantified S-formulas. Equivalence of two formulas, as usual, means that an assignment to the free variables either satisfies or falsifies both formulas.

Problem:	$\mathsf{QEQUIV}_k(S)$
Input:	two $\mathsf{QCSP}_k(S)$-formulas φ and ψ with free variables $z_1, \dots, z_n$
Question:	Is $\varphi[z_1/a_1, \dots, z_n/a_n] = \psi[z_1/a_1, \dots, z_n/a_n]$ for every $a_1, \dots, a_n \in \mathcal{D}$?

As before, omitting the index, $\mathsf{QEQUIV}(S)$ denotes the problem with unbounded number of quantifier alternations.

As for the two problems before, we start with the question if the Galois connection holds. However, this time the situation is more complicated than before.

Proposition 4.10. *Let S_1 and S_2 be constraint languages over the same domain $\mathcal{D}$ such that* $\mathrm{Pol}(S_2) \subseteq \mathrm{Pol}(S_1)$. *Then*

$$\mathsf{QEQUIV}(S_1) \leq_m^{\log} \mathsf{QEQUIV}(S_2 \cup \{=^{\mathcal{D}}\}) \text{ and } \mathsf{QEQUIV}_k(S_1) \leq_m^{\log} \mathsf{QEQUIV}_k(S_2 \cup \{=^{\mathcal{D}}\})$$

for all k.

Proof. Immediate consequence of Proposition 2.3. □

As for satisfiability and model checking, we would like to state a version of the above proposition, where we conclude $\mathsf{QEQUIV}_k(S_1) \leq_m^{\log} \mathsf{QEQUIV}_k(S_2)$. Thus we have to get rid of the new equality clauses introduced by applying Proposition 2.3. In Propositions 3.2 and 4.4 we reach this by simply identifying those variables that are connected by equality clauses. Here, however, we would change the set of satisfying assignments by this, leading to a formula no longer equivalent to the original one, and therefore not obtain the desired reduction. The only way to remove the equality clauses is to express them using relations from S_2. A formal definition of this idea, which will also be useful for us here, is given in [ABI$^+$05] as follows:

A constraint language S *can express* the relation $R(x_1, ..., x_n)$ if there is an S-formula $R_1(z^1_1, \dots, z^1_{n_1}) \wedge \dots \wedge R_l(z^l_1, \dots, z^l_{n_l})$, with $z^i_j \in \{x_1, \dots, x_n, y_1, \dots, y_r\}$ (the z^i_j's need not be distinct) such that for each assignment of values $(c_1, \dots, c_n)$ to the variables $x_1, \dots, x_n$, $R(c_1, ..., c_n)$ evaluates to true if and only if there is an assignment of values to the variables $y_1, \dots, y_r$ such that all R_i-clauses, with x_i replaced by c_i, evaluate to true. If R is the equality relation, we also simply say *S can express equality.*

If the "larger" constraint language can express equality, we obtain the following stronger statement of Proposition 4.10:

Proposition 4.11. *Let S_1 and S_2 be constraint languages over the same domain such that S_2 can express equality and* $\mathrm{Pol}(S_2) \subseteq \mathrm{Pol}(S_1)$. *Then*

$$\mathsf{QEQUIV}(S_1) \leq^{\log}_m \mathsf{QEQUIV}(S_2) \text{ and } \mathsf{QEQUIV}_k(S_1) \leq^{\log}_m \mathsf{QEQUIV}_k(S_2)$$

for all k.

Proof. Apply Proposition 4.10 and then replace the equality clauses by S_2-formulas expressing equality. The new variables introduced in this last step will be existentially quantified in the last quantifier block. □

In the case of equivalence problems, besides the Galois connection we use a second powerful tool, that of *duality.*

Lemma 4.12. *Let S_1 and S_2 be constraint languages that can express equality such that* $\mathrm{Pol}(S_1) = \mathrm{dual}(\mathrm{Pol}(S_2))$. *Then* $\mathsf{QEQUIV}(S_1) \equiv \mathsf{QEQUIV}(S_2)$.

Proof. For a constraint relation R, let $\overline{R} := \{\overline{m} \mid m \in R\}$ where the tuple $\overline{m}$ is obtained from tuple m by componentwise complementation, and for a constraint language S, let $\overline{S} := \{\overline{R} \mid R \in S\}$. Then $\mathrm{Pol}(\overline{S}) = \mathrm{dual}(\mathrm{Pol}(S))$. Thus it is sufficient, because of the Galois connection, to show $\mathsf{QEQUIV}(S) \leq^{\log}_m \mathsf{QEQUIV}(\overline{S})$.

Let φ_1, φ_2 be CSP(S)-formulas. Let $\overline{\varphi_1}, \overline{\varphi_2}$ be the formulas derived from φ_1 and φ_2 by exchanging every occurring relation R with the corresponding relation $\overline{R}$. We claim $\varphi_1 \equiv \varphi_2$ implies $\overline{\varphi_1} \equiv \overline{\varphi_2}$ (the other direction follows for symmetry reasons). It suffices to show $\overline{\varphi_1} \Rightarrow \overline{\varphi_2}$.

Let $a_1, \dots, a_n \in \{0, 1\}$ such that $\overline{\varphi_1}[x_1/a_1, \dots, x_n/a_n]$ holds. Then $\varphi_1[x_1/\overline{a_1}, \dots, x_n/\overline{a_n}]$ holds, and since $\varphi_1 \equiv \varphi_2$, this implies that $\varphi_2[x_1/\overline{a_1}, \dots, x_n/\overline{a_n}]$ holds as well. Therefore, we conclude that $\overline{\varphi_2}[x_1/a_1, \dots, x_n/a_n]$ holds. □

In general, the equivalence problem for quantified Boolean formulas is PSPACE-complete. More precisely, we have the following upper bounds:

Lemma 4.13.
1. *For any Boolean constraint language S and $k \geq 0$,* $\mathsf{QEQUIV}(S) \in$ PSPACE *and* $\mathsf{QEQUIV}_k(S) \in \Pi_{k+1}$P.
2. *If S is a constraint language that is Schaefer, then* $\mathsf{QEQUIV}(S) \in$ coNP.

Proof. This follows by making a case distinction whether k is odd or even, and by expressing equivalence under all assignments using an outermost universal quantifier as follows: $\varphi(x_1, \dots, x_n) \equiv \psi(x_1, \dots, x_n)$ if and only if

$$\forall x_1, \dots, x_n \big(\varphi(x_1, \dots, x_n) \wedge \psi(x_1, \dots, x_n)) \vee (\neg\varphi(x_1, \dots, x_n) \wedge \neg\psi(x_1, \dots, x_n)\big).$$

□

The following lemma provides lower bounds:

Lemma 4.14. *For any Boolean constraint language S and all $k \geq 0$, we have $\overline{\mathsf{QCSP}_{k+1}(S)} \leq_m^{\log}$ $\mathsf{QEQUIV}_k(S)$.*

Proof. Let k be odd, and let $\varphi = \forall X_1 \ldots \exists X_{k+1}\psi$. Then φ is true (and thus not in $\mathsf{QCSP}_{k+1}(S)$, since $k+1$ is even), if and only if $\exists X_2 \ldots \exists X_{k+1}\psi$ is equivalent to 1 (1 can be expressed by fully existentially quantifying a non-empty relation).

Now, let k be even, and let $\varphi = \exists X_1 \forall X_2 \ldots \exists X_{k+1}$ be an instance of $\overline{\mathsf{QCSP}_{k+1}(S)}$. Observe that φ is false (and thus not in $\mathsf{QCSP}_{k+1}(S)$, since $k+1$ is odd) if and only if $\forall X_2 \ldots \exists X_{k+1}$ is equivalent to 0. □

These lemmas together with Theorem 4.3 lead to the following hardness results.

Proposition 4.15. *Let S be a Boolean constraint language with $\mathrm{Pol}(S) \subseteq \mathrm{N}$, and let $k \geq 0$. Then $\mathsf{QEQUIV}_k(S)$ is complete for $\Pi_{k+1}\mathrm{P}$.*

Next we turn to cases with lower complexity, not depending on the quantifier depth. We first identify tractable cases of the equivalence problem.

Theorem 4.16. *Let S be a Boolean constraint language. If S is affine, bijunctive, IHSB$-$, or IHSB$+$, then $\mathsf{QEQUIV}(S)$ is solvable in polynomial time.*

Proof. The main idea of the proof is to construct, for two given formulas, equivalent ones where quantification does not occur. For affine constraint languages, this first step can be performed in polynomial time mainly in using Gaussian elimination (see [CKS01] for a detailed algorithm). For the bijunctive and IHSB cases the two equivalent formulas are obtained by means of Q-resolution (see [KL99, Theorem 7.4.6]). Observe that it is here fundamental that the IHSB- constraint language be finite in order to get only a polynomial number of possible resolvents, thus insuring that this step can be performed in polynomial time.

Thus we have reduced the quantified equivalence problem to the unquantified one, which is in P for the cases examined here (see [BHRV02]). □

Another result from Kleine-Büning and Lettmann [KL99, Theorem 7.5.4] classifies the complexity for Horn-formulas. Translated into our vocabulary, and using the duality of the clones, this gives us (note that membership in coNP follows from above):

Proposition 4.17. *Let S be a Boolean constraint language such that $\mathrm{Pol}(S) \in \{\mathrm{E}_2, \mathrm{V}_2\}$. Then $\mathsf{QEQUIV}(S)$ is coNP-complete, and, in fact, already $\mathsf{QEQUIV}_1(S)$ is coNP-hard.*

We can generalize this to include the constant polymorphisms as well, with a construction similar to the one allowing all essentially unary functions in Lemma 4.2. This gives us:

Theorem 4.18. *Let S be a Boolean constraint language such that $\mathrm{Pol}(S) \subseteq \mathrm{E}$ or $\mathrm{Pol}(S) \subseteq \mathrm{V}$. Then $\mathsf{QEQUIV}_k(S)$ is coNP-hard for all $k \geq 1$.*

Proof. We prove the theorem for the case $\mathrm{Pol}(S) \subseteq \mathrm{E}$. The dual case $\mathrm{Pol}(S) \subseteq \mathrm{V}$ then follows from Lemma 4.12, since constraint languages with these sets of polymorphisms always can express equality due to [ABI+05]. Let S be a constraint language such that $\mathrm{Pol}(S) = \mathrm{E}_2$. We show that $\mathsf{QEQUIV}_1(S) \leq_m^{\log} \mathsf{QEQUIV}_k(S')$ for some constraint language S' for which $\mathrm{M} \supseteq \mathrm{Pol}(S') \supseteq \mathrm{E}$ holds. The result then follows from Proposition 4.17 and the Galois connection stated in Proposition 4.11. Note that, by [ABI+05], if $\mathrm{Pol}(S) \subseteq \mathrm{M}$ then S can express equality.

For an n-ary relation $R \in S$, we construct an $n+2$-ary relation R which can be used to express R for our purposes, and which has both constant polymorphisms. We define

$$R' = \{(u, v, x_1, \dots, x_n \mid (x_1, \dots, x_n) \in R \text{ or } u = v = x_1 = \dots = x_n\}.$$

It is obvious that R' is closed under both constant polymorphisms and under conjunction.

Let S' be defined as $\{R' \mid R \in S\} \cup \{\rightarrow\}$. As mentioned above, it suffices to prove hardness for this special choice of S'. The relation of logical implication, $\rightarrow$, can easily be seen to be invariant under conjunction, and obviously it contains both constant tuples. Hence we know that $\mathrm{Pol}(S') \supseteq \mathrm{E}$, and since $\mathrm{Pol}(\rightarrow) = \mathrm{M}$, we know that $\mathrm{M} \supseteq \mathrm{Pol}(S') \supseteq \mathrm{E}$ holds, as required. Now let φ_1 and φ_2 be $\mathsf{QCSP}_1(S)$-formulas with the same set of free variables, i.e., let

$$\varphi_1 = \exists x_1, \dots, x_n \bigwedge_{i=1}^{l_1} R_i^1(u_1^i, \dots, u_{r_i}^i),$$
$$\varphi_2 = \exists y_1, \dots, y_m \bigwedge_{i=1}^{l_2} R_i^2(v_1^i, \dots, v_{s_i}^i),$$

where the occurring R_i^1 (R_i^2) are r_i-ary (s_i-ary, resp.) relations from S, the occurring variables u_t^i are either from $\{x_1, \dots, x_n\}$, or from the set $\{z_1, \dots, z_k\}$ of free variables of φ_1, and similarly the variables v_t^i are from $\{y_1, \dots, y_m, z_1, \dots, z_k\}$.

We define formulas ψ_1 and ψ_2 as follows:

$$\psi_1' = \exists x_1, \dots, x_n \bigwedge_{i=1}^{l_1} R_i'^1(a, b, u_1^i, \dots, u_{r_i}^i),$$
$$\psi_2' = \exists y_1, \dots, y_m \bigwedge_{i=1}^{l_2} R_i'^2(a, b, v_1^i, \dots, v_{s_i}^i),$$

where a, b are additional free variables. Finally we obtain ψ_1 and ψ_2 by adding, for each variable z free in ϕ_1 (ϕ_2, resp.), the clause $z \rightarrow a$ and the clause $b \rightarrow z$ to the above formulas. Then, by definition, ψ_1 and ψ_2 are $\mathsf{QCSP}_k(S')$-formulas.

The relationship between these formulas is stated in the following claim which is easy to check:

Claim. For an assignment I to the free variables of ψ_1, I satisfies ψ_1 if and only if one of the following two conditions applies:

1. I is constant, or
2. I restricted to the variables appearing in φ_1 satisfies φ_1, and $I(a) = 1$, $I(b) = 0$.

An analogous claim obviously holds for the formulas φ_2 and ψ_2. We now show that φ_1 and φ_2 are equivalent if and only if ψ_1 and ψ_2 are.

First, assume that φ_1 and φ_2 are equivalent, and let I be some assignment to the free variables of ψ_1 such that $I \models \psi_1$. We show that I is also a solution for ψ_2, the equivalence of the formulas then follows due to symmetry. Due to the claim above, we have two cases to consider: if I is a constant assignment, then, by the analogous claim for ψ_2, we know that I is a solution for ψ_2. Otherwise, due to the claim above, we know that $I(a) = 1$, $I(b) = 0$, and I restricted to the variables appearing in φ_1 is a solution to the latter formula. Since φ_1 and φ_2 are equivalent, this implies that the restriction of I is a solution for φ_2 as well, and due to the analogous claim for ψ_2, it follows that I is a solution of ψ_2.

Now assume that ψ_1 and ψ_2 are equivalent, and let I be a solution for φ_1. Again, due to symmetry, it suffices to show that I is a solution for φ_2 as well. Due to the claim above, we know that the solution I' obtained from I by augmenting it with the assignments $I'(a) = 1$, and $I'(b) = 0$, is a solution for ψ_1. Since ψ_1 and ψ_2 are equivalent, this implies that I' is also a solution for ψ_2. Therefore, due to the analogous claim for ψ_2, it follows that I is a solution for φ_1, as claimed. □

Thus, we obtain a full classification of the equivalence problem for quantified Boolean constraint satisfaction problems.

Theorem 4.19. *Let S be a Boolean constraint language.*

1. *If S is affine, bijunctive,* IHSB− *or* IHSB+*, then* $\mathsf{QEQUIV}(S)$ *is in* P*;*
2. *else, if S is Horn or dual Horn, then* $\mathsf{QEQUIV}_k(S)$ *is in* P *for $k = 0$, while* $\mathsf{QEQUIV}_k(S)$ *for $k > 0$ and* $\mathsf{QEQUIV}(S)$ *are* coNP*-complete;*
3. *otherwise* $\mathsf{QEQUIV}_k(S)$ *is complete for* $\Pi_{k+1}\mathrm{P}$ *for each $k > 0$ and* $\mathsf{QEQUIV}(S)$ *is* PSPACE-*complete.*

Proof. As mentioned in Sect. 2, a constraint language S is affine iff $\mathrm{Pol}(S) \supseteq \mathrm{L}_2$, S is bijunctive iff $\mathrm{Pol}(S) \supseteq \mathrm{D}_2$, S is IHSB− iff $\mathrm{Pol}(S) \supseteq \mathrm{S}_{10}$, and S is IHSB+ iff $\mathrm{Pol}(S) \supseteq \mathrm{S}_{00}$. The cases considered in Proposition 4.15, Theorem 4.16, Theorem 4.18 thus cover the whole lattice and we have reached the full classification. The results for $k = 0$ are given in [BHRV02]. □

5 Counting Problems for Quantified Boolean CSPs

5.1 Introduction to Counting Problems and Reductions

Let Σ, S be alphabets and let $R \subseteq \Sigma^* \times S^*$ be a binary relation between strings such that, for each $x \in \Sigma^*$, the set $R(x) = \{y \in S^* \mid R(x, y)\}$ is finite. We write $\#R$ to denote the following counting problem: Given a string $x \in \Sigma^*$, find the cardinality $|R(x)|$ of the set $R(x)$ associated with x.

Valiant [Val79a,Val79b] was the first to investigate the computational complexity of counting problems. To this end, he introduced the class #P of counting functions that count the number of accepting paths of nondeterministic polynomial-time Turing machines. Toda [Tod91a,Tod91b] (cf. also [Vol94,HV95]) introduced higher complexity counting classes using a predicate-based framework that focuses on the complexity of membership in the witness sets. Specifically, if $\mathcal{C}$ is a complexity class of decision problems, then $\#{\cdot}\mathcal{C}$ is the class of all counting problems whose witness relation R satisfies the following conditions:

1. There is a polynomial $p(n)$ such that for every x and every y with $R(x, y)$, we have that $|y| \le p(|x|)$, where $|x|$ is the length of x and $|y|$ is the length of y.
2. The witness recognition problem "given x and y, does $R(x, y)$ hold?" is in $\mathcal{C}$.

Following Toda [Tod91a,Tod91b,Vol94,HV95], $\#{\cdot}\Sigma_k\mathrm{P} \subseteq \#{\cdot}\Pi_k\mathrm{P} = \#\mathrm{P}^{\Sigma_k\mathrm{P}} \subseteq \#{\cdot}\Sigma_{k+1}\mathrm{P}$ holds for each k.

Several notions of reducibilities among counting problems have been defined. The strongest is the one of *parsimonious reduction* [Val79a], which is a polynomial-time many-one reduction preserving the number of witnesses, i.e., $\#A$ reduces to $\#B$ by parsimonious reductions ($\#A \le^{\mathrm{p}}_{par} \#B$) if there is a polynomial-time computable function g such that $|A(x)| = |B(g(x))|$ for all strings x. The aforementioned counting classes are closed under this reduction, #SAT is complete for #P under these reductions, but not many more completeness results are known since the reductions are very strict. Valiant himself obtained only very few completeness results for #P under parsimonious reductions besides the one for #SAT. For most of his other results, in particular for the permanent, he used *Turing reductions*: $\#A$ reduces to $\#B$ by Turing reductions ($\#A \le^{\mathrm{p}}_{T} \#B$) if $\#A$ can be computed by a polynomial-time oracle Turing machine with oracle $\#B$. For most of Valiant's hardness proofs, even Turing-reductions with one oracle query ($\le^{\mathrm{p}}_{1\text{-}T}$) are sufficient. Zankó [Zan91] refined Valiant's results and introduced *counting reductions* (essentially truth-table-reductions with one oracle query), i.e., $\#A$ reduces to $\#B$ by counting reductions ($\#A \le^{\mathrm{p}}_{cnt} \#B$) if there exist polynomial-time computable functions f, g such that $\#A(x) = f\big(\#B(g(x))\big)$ for all strings x. She proved that the permanent is complete for #P not only under Turing-reductions (as shown by

Valiant [Val79a]) but even under counting reductions. In fact, today many problems complete for #P under counting reductions are known. However, the aforementioned counting classes are not closed under these reductions. In fact, the closure of #P under counting reductions gives already $\#\cdot\mathrm{PH}$ [TW92], where $\mathrm{PH} = \bigcup_i \Sigma_i\mathrm{P}$.

Thus, when studying counting problems related to classes of the form $\#\cdot\Sigma_i\mathrm{P}$, researchers have looked for reductions that are powerful enough to obtain many completeness proofs but strict enough to be able to distinguish between different classes $\#\cdot\Sigma_k\mathrm{P}$. Durand, Hermann, and Kolaitis [DHK05] have introduced *subtractive reductions*, defined as follows: We say that $\#A$ reduces to $\#B$ via a strong subtractive reduction, if there exist polynomial-time computable functions f, g such that for every string x, $B(g(x)) \subseteq B(f(x))$ and $|A(x)| = |B(f(x))| - |B(g(x))|$. Subtractive reductions ($\leq^{\mathrm{P}}_{sub}$) now are the transitive closure of strong subtractive and parsimonious reductions.

Durand et al. showed that all classes of the form $\#\cdot\Pi_k\mathrm{P}$ are closed under subtractive reductions, but the closure of $\#\cdot\Sigma_k\mathrm{P}$ under subtractive reductions is $\#\cdot\Pi_k\mathrm{P}$. This is maybe not as nice as one would have wished, since not all of Toda's counting classes are closed; however, for most applications it is sufficient, because, if a problem is complete for $\#\cdot\Pi_k\mathrm{P}$ under subtractive reductions it cannot be in $\#\cdot\Pi_{k-1}\mathrm{P}$ unless $\#\cdot\Pi_k\mathrm{P} = \#\cdot\Pi_{k-1}\mathrm{P}$ (which implies a collapse of the polynomial-time hierarchy to $\Sigma_k\mathrm{P}$ [Vol94]), and a problem complete for $\#\cdot\Sigma_k\mathrm{P}$ cannot be in $\#\cdot\Sigma_{k-1}\mathrm{P}$ unless $\#\cdot\Sigma_k\mathrm{P} = \#\cdot\Pi_{k-1}\mathrm{P}$ (which implies that $\mathrm{UP}^{\Sigma_{k-1}\mathrm{P}} = \Sigma_k\mathrm{P}$ [Vol94], i.e., that $\Sigma_k\mathrm{P}$ can be made "unambiguous" [HV95], which is considered unlikely).

Hence, under reasonable complexity-theoretic assumptions, subtractive reductions can distinguish between the different levels of the $\#\cdot\Sigma_k\mathrm{P}$-hierarchy. Durand et al. obtained many completeness results in this vein, concerning circumscription and related non-monotonic logics.

In [BCC$^+$05] the notion of subtractive reductions was generalized to *complementive reduction*, which appeared to be useful for Boolean constraint satisfaction problems involving complementive relations. This reduction is also suitable for the counting problems we consider in this section.

Before we can state the definition of complementive reductions, we need some additional notions. We enlarge every permutation π on an alphabet Γ to the strings in Γ^* by means of $\pi(a_1 \cdots a_k) = \pi(a_1) \cdots \pi(a_k)$ for each string $a_1 \cdots a_k \in S^*$. A set of strings $E \subseteq \Gamma^*$ over an alphabet Γ is called *complementive* if there is a permutation π on Γ such that for all x, if $x \in E$ then $\pi(x) \in E$.

Definition 5.1. Let $\Sigma_1, \Sigma_2, \Gamma_1, \Gamma_2$ be alphabets and let $\#A$ and $\#B$ be two counting problems determined by the binary relations $A \subseteq \Sigma_1^* \times \Gamma_1^*$ and $B \subseteq \Sigma_2^* \times \Gamma_2^*$.

We say that $\#A$ reduces to $\#B$ via a *strong complementive reduction*, if there exist polynomial-time computable functions $f, g\colon \Sigma_1^* \to \Sigma_2^*$ such that for every string $x \in \Sigma_1^*$:

– $B(x)$ is complementive,
– $B(g(x)) \subseteq B(f(x))$,
– $2 \cdot |A(x)| = |B(f(x))| - |B(g(x))|$.

A *complementive reduction* $\#A \leq^{\mathrm{P}}_{compl} \#B$ is a sequence of strong complementive, strong subtractive, or parsimonious reductions.

The need for complementive reductions arises in our context because the sets of satisfying assignments for formulas φ built over, e.g., a NAE^2-relation is invariant under negation. In particular, this implies that the number of satisfying assignments of φ will always be a multiple of 2. Therefore, a parsimonious reduction from arbitrary problems in the counting hierarchy is not possible.

The following theorem from [BCC$^+$05] shows that the counting classes share the same closure properties under complementive reductions as under subtractive reductions.

Theorem 5.2 ([BCC$^+$05]). *The class #P and all higher complexity classes $\#\cdot\Pi_k\mathrm{P}$, $k \geq 1$, are closed under complementive reductions.*

Unfortunately, the $\#\cdot\Sigma_k$P-classes are not closed under complementive reductions (unless $\#\cdot\Sigma_k\text{P} = \#\cdot\Pi_k\text{P}$, see [BCC+05], which implies that the polynomial-time hierarchy collapses to Σ_kP [Vol94]). Nevertheless, these reductions are suitable to distinguish between the levels of the $\#\cdot\Sigma_k$P-hierarchy. We will prove a number of completeness results under complementive reductions for classes of the form $\#\cdot\Sigma_k$P; and as for subtractive reductions we then know that these problems cannot be in $\#\cdot\Sigma_{k-1}$P unless $\#\cdot\Sigma_k\text{P} = \#\cdot\Pi_{k-1}\text{P}$

5.2 Counting the Number of Satisfying Assignments

In the Boolean case Creignou and Hermann [CH96] proved that the complexity of the counting problem #SAT(S) of S-formulas is dichotomous: #SAT(S) is in FP (the class of all functions computable in polynomial time), if S is a set of affine relations, otherwise #SAT(S) is #P-complete under counting reductions. Bauland et al. [BCC+05] exhibited a trichotomy result for the counting problem associated with conjunctive queries, i.e., existentially quantified formulas, denoted by $\#\mathsf{SAT\text{-}COQ}$: $\#\mathsf{SAT\text{-}COQ}(S)$ is in FP if S is affine, else $\#\mathsf{SAT\text{-}COQ}(S)$ is #P-complete under counting reductions if S is bijunctive, Horn, or dual Horn, and otherwise $\#\mathsf{SAT\text{-}COQ}(S)$ is $\#\cdot$NP-complete under complementive reductions.

We are interested in the counting problem associated with $\mathsf{QCSP}_k(S)$-formulas. Let us first look at the case of unrestricted quantified propositional formulas (not necessarily CNF or S-formulas for some S). Here, given a formula φ with free variables Y, $\varphi(Y) = \exists X_1 \forall X_2 \ldots Q_k X_k \psi(Y, X_1, \ldots, X_k)$, where ψ is quantifier-free, we are interested in the number of assignments for Y such that $\varphi(Y)$ holds. We will denote this number by $\#\mathsf{sat}(\varphi)$ (and by $\#\mathsf{unsat}(\varphi)$ the number of assignments for Y such that $\varphi(Y)$ does not hold). Let us denote by $\#\mathsf{QSAT}_k$ the problem of counting the satisfying assignments of a quantified Boolean formula with free variables and $k-1$ quantifier alternations starting with an $\exists$-quantifier. This problem is prototypical for $\#\cdot\Sigma_k$P-complete problems under parsimonious reductions. It remains $\#\cdot\Sigma_k$P-complete when the formula is restricted to be 3-CNF for i odd, and 3-DNF for i even, as shown in [DHK05] building on the results by Wrathall [Wra77] explained already in Sect. 3.

The following problems are the counting versions of the decision problems studied in Section 4.1. Let S be a finite set of logical relations.

Problem:	$\#\mathsf{QCSP}_k(S)$
Input:	a $\mathsf{QCSP}_k(S)$-formula φ with free variables
Output:	if k is odd: $\#\mathsf{sat}(\varphi)$;
	if i is even: $\#\mathsf{unsat}(\varphi)$

Observe that $\#\mathsf{QCSP}_1(S)$ is the same as the problem $\#\mathsf{SAT\text{-}COQ}(S)$ studied in [BCC+05]. Note that $\#\mathsf{QCSP}_i(S) \in \#\cdot\Sigma_i\text{P}$, and that according to the remark above $\#\mathsf{QCSP}_i(\text{S}_{\mathsf{3SAT}})$ is $\#\cdot\Sigma_i$P-complete under parsimonious reductions. Our goal is to study the complexity of $\#\mathsf{QCSP}_i(S)$ for all possible sets S. A central result for our development is the following easy consequence of Proposition 2.3. It states that the Galois connection holds for the counting problem, with a proof identical to the one of Proposition 3.2.

Proposition 5.3. *Let S_1 and S_2 be constraint languages over the same domain. If the inclusion* $\text{Pol}(S_2) \subseteq \text{Pol}(S_1)$ *holds, then there exists a parsimonious reduction from* $\#\mathsf{QCSP}_k(S_1)$ *to* $\#\mathsf{QCSP}_k(S_2)$*, for any $k \geq 1$.*

Our work will essentially follow the same line as the one for the corresponding decision problems in Sect. 4.1.

Lemma 5.4. $\#\mathsf{QCSP}_k(\text{R}_{1/3})$ *is $\#\cdot\Sigma_k$P-complete under parsimonious reductions, for any $k \geq 1$.*

Proof. $\#\mathsf{QCSP}_k(\text{S}_{\mathsf{3SAT}})$ is $\#\cdot\Sigma_k$P-complete under parsimonious reductions. Now apply Proposition 5.3 (remember $\text{Pol}(\text{R}_{1/3}) = \text{I}_2$). □

Lemma 5.5. $\#\mathsf{QCSP}_k(\mathrm{NAE}^2)$ *is* $\#\cdot\Sigma_k\mathrm{P}$*-complete under complementive reductions, for any* $k \geq 1$.

Proof. We show that $\#\mathsf{QCSP}_k(\mathrm{R}_{1/3})$ can be reduced to $\#\mathsf{QCSP}_k(\mathrm{NAE}^2)$. The construction is very similar to the one in the proof of Proposition 4.1.

Let $\varphi(Y)$ be a $\mathsf{QCSP}_k(\mathrm{R}_{1/3})$-formula with free variables Y (suppose $Y = \{y_1, \dots, y_n\}$) and $\varphi(Y) = Q_k X_1 \dots \forall X_{k-1} \exists X_k C_1 \wedge \cdots \wedge C_m$ such that each C_j is of the form $C_j = \mathrm{R}_{1/3}(v_{j_1}, v_{j_2}, v_{j_3})$ for some $v_{j_1}, v_{j_2}, v_{j_3} \in Y \cup X_1 \cup \cdots \cup X_i$. Consider now the formula

$$\varphi_1(Y, u, v) = Q_k X_1 \dots \forall X_{k-1} \exists X_k \bigwedge_{j=1}^{m} C_j \wedge \mathrm{R}_{1/3}(u, u, v),$$

where u and v are two new variables. Observe that $\#\mathsf{sat}(\varphi_1) = \#\mathsf{sat}(\varphi)$ and $\#\mathsf{unsat}(\varphi_1) = 2^{n+2} - \#\mathsf{sat}(\varphi)$. Now, let t be an additional new variable, and construct the formula $\varphi_2(Y, u, v) = Q_k X_1 \dots \forall X_{k-1} \exists X_k \exists t \bigwedge_{j=1}^{m} \mathrm{R}_{2/4}(v_{j_1}, v_{j_2}, v_{j_3}, t) \wedge \mathrm{R}_{2/4}(u, u, v, t)$, where each relation $\mathrm{R}_{2/4}(a, b, c, d)$ stands for the equivalent conjunction of NAE^2-clauses. We get a $\mathsf{QCSP}_k(\mathrm{NAE}^2)$-formula $\varphi_2(Y, u, v)$ such that $\#\mathsf{sat}(\varphi_2) = 2\#\mathsf{sat}(\varphi)$ and $\#\mathsf{unsat}(\varphi_2) = 2^{n+2} - 2\#\mathsf{sat}(\varphi)$.

Now consider the formula $\varphi_3(Y, u, v) = \mathrm{NAE}^2(u, u, v) \wedge \bigwedge_{j=1}^{n} \mathrm{NAE}^2(u, v, y_j)$. Observe that $\mathsf{unsat}(\varphi_3) \subseteq \mathsf{unsat}(\varphi_2)$, and that $\#\mathsf{unsat}(\varphi_3) = 2^{n+1}$. For k odd, we use $\varphi_2(Y, u, v)$ constructed from $\varphi(Y)$ above. This is a $\mathsf{QCSP}_k(\mathrm{NAE}^2)$-formula which verifies $\#\mathsf{sat}(\varphi) = \#\mathsf{sat}(\varphi_2)/2$. For k even we construct the pair $(\varphi_2(Y, u, v), \varphi_3(Y, u, v))$ of $\mathsf{QCSP}_k(\mathrm{NAE}^2)$-formulas, which verify $\mathsf{unsat}(\varphi_3) \subseteq \mathsf{unsat}(\varphi_2)$ and $\#\mathsf{unsat}(\varphi) = \frac{\#\mathsf{unsat}(\varphi_2) - \#\mathsf{unsat}(\varphi_3)}{2}$. Thus, in both cases we have a complementive reduction from $\#\mathsf{QCSP}_k(\mathrm{R}_{1/3})$ to $\#\mathsf{QCSP}_k(\mathrm{NAE}^2)$. □

Lemma 5.6. *There exists a Boolean relation* R_0 *such that* $\mathrm{N} \subseteq \mathrm{Pol}(R_0)$ *and* $\#\mathsf{QCSP}_k(R_0)$ *is* $\#\cdot\Sigma_k\mathrm{P}$*-complete under complementive reductions, for any* $k \geq 1$*..*

Proof. Observe that the reduction provided in the proof of Lemma 4.2 is parsimonious. Thus, for $k \geq 2$ the conclusion follows from Lemma 5.5. The case $k = 1$ follows from [BCC+05]. □

We are now in a position to prove the following complexity classification, which completely classifies the $\#\mathsf{QCSP}_i(S)$ problem for the Boolean case.

Theorem 5.7. *Let* S *be a Boolean constraint language and* $k \geq 1$.

- *If* S *is affine, then* $\#\mathsf{QCSP}_k(S)$ *is in* FP*,*
- *else if* S *is bijunctive, or Horn, or dual Horn, then* $\#\mathsf{QCSP}_k(S)$ *is* $\#\mathrm{P}$*-complete under counting reductions,*
- *otherwise,* $\#\mathsf{QCSP}_k(S)$ *is* $\#\cdot\Sigma_k\mathrm{P}$*-complete under complementive reductions.*

Proof. If S is affine, then the Gaussian elimination algorithm given in [CH96] for $\#\mathsf{CSP}(S)$ can also be used to construct a corresponding polynomial-time algorithm for $\#\mathsf{QCSP}_i(S)$.

If S is Horn, dual Horn, or bijunctive, then $\mathsf{QCSP}(S)$ (and a fortiori $\mathsf{QCSP}_i(S)$) is in P (see Theorem 3.1) and therefore $\#\mathsf{QCSP}_i(S)$ is in #P. Moreover, we know from [CH96] that in this case $\#\mathrm{SAT}(S)$ is #P-hard. Hence, the trivial reduction from $\#\mathrm{SAT}(S)$ to $\#\mathsf{QCSP}_i(S)$ shows that $\#\mathsf{QCSP}_i(S)$ is #P-complete.

The only remaining case $\mathrm{Pol}(S) = \mathrm{N}$ follows from Lemma 5.6 and Proposition 5.3. □

Our trichotomy for Boolean $\#\mathsf{QCSP}_i(S)$ yields (with the same proofs) a classification of the counting problem in the case of an unbounded number of alternations. Denoting this problem by $\#\mathsf{QCSP}(S)$, a classification completely analogous to Theorem 5.7, but replacing $\#\cdot\Sigma_i\mathrm{P}$ by #PSPACE, is obtained. Here, #PSPACE in the sense of Valiant [Val79a] denotes $\#\mathrm{P}^{\mathrm{PSPACE}}$. It is easy to observe that #PSPACE coincides with Ladner's class ♮PSPACE [Lad89]. (*Caveat:* What

Ladner denotes by #PSPACE is a *different class.*) Ladner proves that #PSPACE = $\natural$PSPACE additionally coincides with FPSPACE(poly), the class of all polynomially length-bounded functions computable in polynomial space, and he observes that #QSAT is complete in this class under parsimonious reductions.

Corollary 5.8. *Let S be a Boolean constraint language.*

- *If S is affine, then* $\#\mathsf{QCSP}(S)$ *is in* FP*,*
- *else if S is bijunctive, or Horn, or dual Horn, then* $\#\mathsf{QCSP}(S)$ *is* #P*-complete under counting reductions,*
- *otherwise,* $\#\mathsf{QCSP}(S)$ *is* #PSPACE*-complete under complementive reductions.*

The above classifications involve reductions that are maybe not the most natural ones. However, they suffice for our goal, a classification of the counting problem for Boolean quantified CSPs. If a problem is complete for a class $\#\cdot\Sigma_k\mathrm{P}$ or #PSPACE it cannot be an element in a lower class (under reasonable complexity-theoretic assumptions), as stated in the final corollary in this section.

Corollary 5.9. *Let S be a finite set of logical relations.*

1. *If S is not affine then* $\#\mathsf{QCSP}_k(S) \notin \mathrm{FP}$ *for any* $k \geq 1$*, unless* $\mathrm{FP} = \#\mathrm{P}$ *and* $\mathrm{P} = \mathrm{NP}$.
2. *If S is not Schaefer then* $\#\mathsf{QCSP}_k(S) \notin \#\cdot\Sigma_{k-1}\mathrm{P}$ *for any* $k \geq 1$*, unless* $\#\cdot\Sigma_k\mathrm{P} = \#\cdot\Pi_{k-1}\mathrm{P}$.
3. *If S is not Schaefer then* $\#\mathsf{QCSP}(S) \notin \#\cdot\Sigma_k\mathrm{P}$ *for any* $k \geq 1$*, unless* $\#\cdot\Pi_k\mathrm{P} = \mathrm{FPSPACE(poly)}$ *and the polynomial-time hierarchy collapses.*

6 Non-Boolean Domains

In the preceding two sections we have proven complete complexity classifications for the problems of evaluation, model checking, equivalence, and counting the number of satisfying assignments for quantified Boolean CSPs. In this section, we will turn to finite domains of arbitrary higher cardinality. We will essentially show how the hardness results from above can be transferred to this case.

6.1 Decision Problems

Let us first turn to the problem of evaluating a quantified CSP. Similarly to Lemma 4.1, it can be shown that the presence of the constraint NAE over a domain of an arbitrary size m leads to a hardness result for the evaluation problem:

Lemma 6.1. $\mathsf{QCSP}_k(\mathrm{NAE}^m)$ *is complete for* $\Sigma_k\mathrm{P}$ *under logspace reductions, for any* $k \geq 1$.

Proof. The proof follows directly from the proof for Lemma 6.6 below, where the even stronger result for the complexity of the corresponding counting result is shown. □

This lemma allows us to identify a larger class of $\Sigma_i\mathrm{P}$-complete problems over finite domains, namely the ones for which the set of polymorphisms consists only of constants or essentially unary functions. A k-ary function $f\colon \mathcal{D}^k \to \mathcal{D}$ is *essentially unary* if there is a non-constant unary function $g\colon \mathcal{D} \to \mathcal{D}$ and some $1 \leq i \leq k$ such that $f(v_1, \ldots, v_k) = g(v_i)$ for all $v_1, \ldots, v_k \in \mathcal{D}$. Again, the construction is similar to the one in the Boolean case (see Lemma 4.2).

Lemma 6.2. *For every finite domain $\mathcal{D}$ with $|\mathcal{D}| = m$, there exists a relation R_0 defined over $\mathcal{D}$ such that* $\mathrm{Pol}(R_0)$ *contains all essentially unary functions and all constants, and such that* $\mathsf{QCSP}_k(\mathrm{NAE}^m)$ *reduces to* $\mathsf{QCSP}_k(R_0)$ *under logspace reductions, for any* $k \geq 2$.

Proof. Let $\mathcal{D}$ be a finite domain of size m. Let R_0 be the $(m+3)$-ary relation

$$R_0 = \big\{(t_1, \dots, t_m, x_1, x_2, x_3) \big| \, |\{t_1, \dots, t_m\}| \leq m-1 \text{ or } \mathrm{NAE}^m(x_1, x_2, x_3)\big\}.$$

It is clear that $\mathrm{Pol}(R_0)$ contains all the constants. It is also easy to see that R_0 is closed under unary functions g (if g is injective, then the NAE-property is invariant under g, and if g is not injective, then $|\{g(t_1), \dots, g(t_m)\}| \leq m-1$), and therefore R_0 is closed under all essentially unary functions on $\mathcal{D}$. Now we prove that $\mathsf{QCSP}_k(\mathrm{NAE}^m)$, which is complete for $\Sigma_k\mathrm{P}$, can be reduced to $\mathsf{QCSP}_k(R_0)$ in logarithmic space.

Let $\varphi = Q_k X_1 \dots \exists X_k \bigwedge_{j=1}^p \mathrm{NAE}^m(x_{j_1}, x_{j_2}, x_{j_3})$ be an instance of $\mathsf{QCSP}_k(\mathrm{NAE}^m)$. Let $\varphi' = Q_k X_1 \dots \forall X_{i-1} \forall t_1 \dots \forall t_m \exists X_k \bigwedge_{j=1}^p R(t_1, \dots, t_m, x_{j_1}, x_{j_2}, x_{j_3})$. It is clear that φ is true if and only if φ' is true, concluding the proof of the lemma. □

Lemma 6.2 now yields the following completeness result.

Theorem 6.3. *Let S be a constraint language over a finite domain of cardinality at least 2, and $k \geq 2$. If $\mathrm{Pol}(S)$ consists only of essentially unary functions and constants, then $\mathsf{QCSP}_k(S)$ is $\Sigma_k\mathrm{P}$-complete and $\mathsf{QCSP}(S)$ is PSPACE-complete under logspace reductions.*

Proof. We have $\mathrm{Pol}(S) \subseteq \mathrm{Pol}(R_0)$, where R_0 is the relation exhibited in Lemma 6.2. Hence, the conclusion follows from Lemma 6.1 and Proposition 3.2. □

As in the Boolean case, the lower bounds for evaluation translate to model checking:

Corollary 6.4. *Let S be a constraint language over a finite domain of cardinality of at least 2, and $k \geq 2$. If $\mathrm{Pol}(S)$ contains only essentially unary functions and constants, then $\mathsf{QMC}_k(S)$ is complete for $\Sigma_k\mathrm{P}$ if k is odd, and complete for $\Pi_k\mathrm{P}$ if k is even, and $\mathsf{QMC}(S)$ is complete for PSPACE under logspace reductions.*

Proof. Follows by the reduction from (the complement) of the evaluation problem for QCSPs to the model checking problem, given in Proposition 4.6, which holds here as well. □

Next we turn to the equivalence problem. Again, the lower bounds for QCSP translate to this case.

Corollary 6.5. *Let S be a constraint language over a finite domain of cardinality of at least 2. If $\mathrm{Pol}(S)$ contains only essentially unary functions and constants, then $\mathsf{QEQUIV}_k(S)$ is complete for $\Pi_{k+1}\mathrm{P}$ and $\mathsf{QEQUIV}(S)$ is complete for PSPACE under logspace-reductions.*

Proof. As in the Boolean domain (see Lemma 4.14) we have for any k that $\overline{\mathsf{QCSP}_{k+1}(S)} \leq \mathsf{QEQUIV}_k(S)$. The corollary then follows from Theorem 6.3. □

6.2 Counting Problems

We have seen that the clone containing all essentially unary or constant functions gives rise to hard constraint satisfaction problems in the decision problem, and in the Boolean counting problem. We now show this hardness result also holds for arbitrary finite domains. Similarly to the Boolean case, we cannot prove hardness under parsimonious reductions for the $\#\mathsf{QCSP}_k$-problem, since in the cases we consider, every permutation of the domain is a polymorphism, and therefore, the number of satisfying solutions of any S-formula will always be a multiple of the size of D. For the Boolean case, complementive reductions were used to solve this problem. The canonical generalization of complementive reductions to arbitrary domains apparently fails to have the property that the relevant counting classes are closed under this generalization.

Therefore, we use a different approach for the non-Boolean case: Instead of counting the solutions themselves, we count the number of equivalence classes of solutions, where two solutions are equivalent if and only if one is obtained by applying a permutation of the domain to the other. The construction in the proof for the following result not only gives a parsimonious reduction among the equivalence classes, but it also yields a reduction which is "almost parsimonious" when counting solutions as usual: The number of solutions of the constructed formula is the number of solutions of the original formula multiplied with a constant, which only depends on the domain. We obtain the following result:

Lemma 6.6. $\#\mathsf{QCSP}_k(\mathrm{NAE}^m)$ *is complete for* $\#{\cdot}\Sigma_k\mathrm{P}$ *under parsimonious reductions among the canonical equivalence classes of the solutions, for any* $m \geq 2$, $k \geq 1$.

Proof. We use ideas from the proof for Proposition 4.1 in [BKBJ02]. Observe that $x \neq y$ can be expressed as $\mathrm{NAE}(x, x, y)$ over any domain. The proof is by induction. The case $m = 2$ follows from Lemma 5.5. We now show $\#\mathsf{QCSP}_k(\mathrm{NAE}^m) \leq^{\mathrm{p}}_{compl} \#\mathsf{QCSP}_k(\mathrm{NAE}^{m+1})$. Let φ be a NAE^m-formula with n free variables $X = \{x_1, \dots, x_n\}$, existentially quantified variables $Y = \{y_1, \dots, y_{n_y}\}$ and universally quantified variables $Z = \{z_1, \dots, z_{n_z}\}$. We add free variables $x_{n+1}, \dots, x_{n+m}$ with inequality constraints between any two of them. We call the result φ_m, and observe that $\#\mathsf{sat}(\varphi_m) = m! \cdot \#\mathsf{sat}(\varphi)$. We construct a formula φ_{m+1} as follows:

- Copy the formula φ_m and replace every relation symbol NAE^m with NAE^{m+1}, and add a new free variable w.
- For each free or $\exists$-quantified variable $v \in X \cup \{x_{n+1}, \dots, x_{n+m}\} \cup Y$, add a constraint $v \neq w$.
- For each universally quantified variable z_i, change $\forall z_i$ to $\forall z'_i$, and add $\exists t_{i,1}, \dots, t_{i,m-1}, z_i$ to the next $\exists$-block, add inequality constraints $t_{i,j} \neq t_{i,k}$ for $j \neq k$ and $(z_i \neq t_{i,j}) \wedge (z'_i \neq t_{i,j}) \wedge (w \neq t_{i,j})$ for all $j \in \{1, \dots, m-1\}$.

Note that the set of satisfying assignments to these formulas is closed under permutations of the domain, and every solution assigns exactly m different values to the free variables $x_1, \dots, x_{n+m}$. Recall that we call two assignments I' and I'' equivalent if there is a permutation Π of the domain such that $I'(x_i) = \Pi(I''(x_i))$ for all i. For each equivalence class I, let I^0 be one canonical representative that does not use the value m, for example the one of minimal lexicographical order. We now show that for these assignments, $I^0 \models \varphi_m$ holds if and only if $(I^0 \cup \{w = m\}) \models \varphi_{m+1}$ holds.

Let $I^0 \models \varphi_m$ and Π_{m+1} be a $\forall$-assignment for φ_{m+1}. Let

$$\Pi_m(z_j) = \begin{cases} \Pi_{m+1}(z'_j) & \text{if } \Pi_{m+1}(z'_j) \in \{0, \dots, m-1\} \\ m-1 & \text{otherwise.} \end{cases}$$

Let E_m be an $\exists$-assignment for φ_m such that $E_m \models \varphi_m(I^0 \cup \Pi_m)$. We define an $\exists$-assignment E_{m+1} for φ_{m+1} as follows:

$E_{m+1}(z_j) = \Pi_m(z_j)$. Assign the $m-2$ different values from $\{0, \dots, m\} \setminus \{m, E_{m+1}(z_j)\}$ to the $t_{j,k}$-variables. This satisfies all inequality constraints between the $t_{j,k}$, those involving w, and the $t_{j,k} \neq z_j$ clauses. The $z'_j \neq t_{j,k}$ clauses are satisfied as well, since z_j is assigned the same value as z'_j unless $\Pi_{m+1}(z'_j) = m$, and in this case the clause is satisfied, because none of the $t_{j,k}$-variables takes the value m.

Let $I_m = I^0 \cup \Pi_m \cup E_m$, and $I_{m+1} = I^0 \cup \{w = m\} \cup \Pi_{m+1} \cup E_{m+1}$. We claim that for any $v \in (X \cup Y \cup Z) \setminus \{w\}$, it holds that $I_m(v) = I_{m+1}(v)$. For free variables $v \in X$, this holds by definition. For an existentially quantified variable $v \in Y$ or a universally quantified variable $v \in Z$, this holds by definition of E_{m+1} (recall that the corresponding variable v is $\exists$-quantified in φ_{m+1}). Thus, the NAE-constraints between variables from $X \cup Y \cup Z$ are satisfied in φ_{m+1}, because they are satisfied in φ_m.

Now, let $I^0 \cup \{w = m\} \models \varphi_{m+1}$, and Π_m be a $\forall$-assignment for φ_m. Let E_{m+1} be an $\exists$-assignment such that $E_{m+1} \models \varphi_{m+1}(I^0 \cup \Pi_m)$. By definition, $I^0 \cup \{w = m\}(w) = m$. Thus, all other variables must take values from $\{0, \dots, m-1\}$, in particular, the z_j and z'_j take the same value in φ_{m+1}. Therefore, the $\exists$-assignment can also be used for φ_m and satisfies the NAE^m-constraints.

Since each I^0 represents $m!$ (resp. $(m+1)!$) satisfying assignments of φ_m (resp. φ_{m+1})—one for each permutation of the domain (note that the value for w is fully determined by the values for $x_{n+1}, \dots, x_{n+m}$ and therefore the additional variable w does not add another factor)—we have

$$\#\mathsf{sat}(\varphi_{m+1}) = (m+1) \cdot \#\mathsf{sat}(\varphi_m) = (m+1)! \cdot \#\mathsf{sat}(\varphi).$$

This gives us a parsimonious reduction, if we consider the number of satisfying (or unsatisfying) assignments up to permutations of the domain. □

Thus we finally obtain the following lower bound for the counting problem over arbitrary finite domains:

Theorem 6.7. *Let S be a constraint language over a finite domain of cardinality of at least 2 such that $\mathrm{Pol}(S)$ only contains constants and essentially unary functions. Then $\#\mathsf{QCSP}_k(S)$ is complete for $\#\cdot\Sigma_k\mathrm{P}$ under parsimonious reductions among the canonical equivalence classes of the solutions, for any $k \geq 1$.*

Proof. $\#\mathsf{QCSP}_k(\mathrm{NAE}^m)$ is complete for $\#\cdot\Sigma_k\mathrm{P}$ due to Lemma 6.6. Now use the same reduction as in Lemma 6.2, which is parsimonious. □

7 Conclusion

In this paper we studied the computational goals of evaluation, model checking, equivalence, and counting for S-formulas/constraint satisfaction problems. In the case of Boolean formulas, we obtained full complexity classifications. It can be seen that when going from satisfiability for unquantied formulas to evaluation/model checking to equivalence to counting, the cases of constraint languages S that admit efficient solutions becomes smaller and smaller, see Fig. 3 (there, satisfiability, i.e., the second column, refers to not quantified S-formulas and Schaefer's Theorem, and the other columns refer to problems for quantified S-formulas and the results of this paper). In the case of general non Boolean domains we obtained a number of hardness results for all our problems.

	Satisfiability	Evaluation and Model Checking	Equivalence	Counting
hard	not Schaefer, not 0-valid, and not 1-valid	not Schaefer	not Schaefer	not Schaefer
not so hard	–	–	Schaefer, not affine, not bijunctive, and not IHSB	Schaefer and not affine
tractable	Schaefer, 0-valid, or 1-valid	Schaefer	affine, bijunctive, IHSB−, or IHSB+	affine

Fig. 3. Summary of complexity results for Boolean constraints

The most obvious remaining open question is of course how to obtain a finer or even complete classification for our problems for non-Boolean universes. A. Bulatov [Bul06] obtained a full classification for satisfiability of (not quantified) CSPs over the 3-element universe – again a dichotomy as in Schaefer's case. Also for the counting problem, progress for not quantified CSP has been made [BD03]. For quantified constraints over arbitrary universes, many complexity results can be found in [BBJK03,Che04a,Che04b,CD05,Che06]. All this may be a hint that for the case of a 3-element universe search for a hopefully complete classification for the four computational goals studied in this paper or at least the counting problem is no hopeless pursuit.

Our results for Boolean S-formulas as summarized in Fig. 3 exhibit a dichotomy for evaluation and model checking and a trichotomy for equivalence and counting *under polynomial-time reductions.* In the Boolean (and maybe also the general) case, a study of the complexity degrees of the here studied problems under stricter reductions such as logspace or even first-order might turn out to be worthwhile. For the satisfiability problem for unquantified CSPs this was done in [ABI+05], and the classification under first-order reductions obtained there together with Agrawal's first-order isomorphism theorem [Agr01] leads to the conclusion that there are only six different satisfiability problems for S-formulas.

To obtain a classification for the counting problem we had to use the conceptionally quite involved complementive reductions. As long as one is only interested in a disctinction between polynomial-time solvable on the one hand side and hard for #P on the other side, the simple counting reductions suffice but, as we saw, these are not able to make fine distinctions in the hierarchy of classes $\#\cdot\Sigma_k$P. The complementive reductions used here leave the $\#\cdot\Pi_k$P-classes closed, but not the $\#\cdot\Sigma_k$P-classes. Though this is sufficient for our purpose here, we want to ask if there is a reduction that yields completeness for interesting counting problem and leaves all the relevant classes closed. But even looking only at the class #P and its "standard" complete problem, the permanent, the more basic question arises if there is a reduction among counting problems under which simultaneously the permanent is complete and #P is closed.

Acknowledgement. We are grateful to Miki Hermann for many helpful comments on different topics of this paper, and to Arnaud Durand for suggesting the last open question in our conclusion.

References

[ABI+05] E. Allender, M. Bauland, N. Immerman, H. Schnoor, and H. Vollmer. The complexity of satisfiability problems: Refining schaefer's theorem. In *Proceedings of the 30th International Symposium on Mathematical Foundations of Computer Science*, pages 71–82, 2005.

[Agr01] M. Agrawal. The first-order isomorphism theorem. In *Foundations of Software Technology and Theoretical Computer Science: 21st Conference, Bangalore, India, December 13-15, 2001. Proceedings*, Lecture Notes in Computer Science, pages 58–69, Berlin Heidelberg, 2001. Springer Verlag.

[BBJK03] F. Börner, A. Bulatov, P. Jeavons, and A. Krokhin. Quantified constraints: algorithms and complexity. In *Proceedings 17th International Workshop on Computer Science Logic*, volume 2803 of *Lecture Notes in Computer Science*, Berlin Heidelberg, 2003. Springer Verlag.

[BCC+05] M. Bauland, P. Chapdelaine, N. Creignou, M. Hermann, and H. Vollmer. An algebraic approach to the complexity of generalized conjunctive queries. In *Proceedings 7th International Conference on Theory and Applications of Satisfiability Testing, Revised Selected Papers*, volume 3542 of *Lecture Notes in Computer Science*, pages 30–45. Springer Verlag, 2005.

[BCRV03] E. Böhler, N. Creignou, S. Reith, and H. Vollmer. Playing with Boolean blocks, part I: Post's lattice with applications to complexity theory. *SIGACT News*, 34(4):38–52, 2003.

[BCRV04] E. Böhler, N. Creignou, S. Reith, and H. Vollmer. Playing with Boolean blocks, part II: Constraint satisfaction problems. *SIGACT News*, 35(1):22–35, 2004.

[BD03] A. Bulatov and V. Dalmau. Towards a dichotomy theorem for the counting constraint satisfaction problem. In *Proceedings Foundations of Computer Science*, pages 562–572. ACM Press, 2003.

[BHRV02] E. Böhler, E. Hemaspaandra, S. Reith, and H. Vollmer. Equivalence and isomorphism for Boolean constraint satisfaction. In *Computer Science Logic*, volume 2471 of *Lecture Notes in Computer Science*, pages 412–426, Berlin Heidelberg, 2002. Springer Verlag.

[BKBJ02] F. Börner, A. Krokhin, A. Bulatov, and P. Jeavons. Quantified constraints and surjective polymorphisms. Technical Report PRG-RR-02-11, Computing Laboratory, University of Oxford, UK, 2002.

[BRSV05] E. Böhler, S. Reith, H. Schnoor, and H. Vollmer. Bases for Boolean co-clones. *Information Processing Letters*, 96:59–66, 2005.

[Bul06] A. A. Bulatov. A dichotomy theorem for constraint satisfaction problems on a 3-element set. *J. ACM*, 53(1):66–120, 2006.

[CD05] H. Chen and V. Dalmau. From pebble games to tractability: An ambidextrous consistency algorithm for quantified constraint satisfaction. In *Proceedings 19th International Workshop on Computer Science Logic*, volume 3634 of *Lecture Notes in Computer Science*, pages 232–247. Springer Verlag, 2005.

[CH96] N. Creignou and M. Hermann. Complexity of generalized satisfiability counting problems. *Information and Computation*, 125:1–12, 1996.

[Che04a] H. Chen. Collapsibility and consistency in quantified constraint satisfaction. In *Proceedings of the Nineteenth National Conference on Artificial Intelligence, Sixteenth Conference on Innovative Applications of Artificial*, pages 155–160. AAAI Press/The MIT Press, 2004.

[Che04b] H. Chen. *The computational complexity of quantified constraint satisfaction.* PhD thesis, Cornell Universtiy, 2004.

[Che06] H. Chen. A rendezvous of logic, complexity, and algebra. *ACM-SIGACT Newsletter*, 37(4):85–114, 2006.

[CKS01] N. Creignou, S. Khanna, and M. Sudan. *Complexity Classifications of Boolean Constraint Satisfaction Problems.* Monographs on Discrete Applied Mathematics. SIAM, 2001.

[CKV07] N. Creignou, Ph. Kolaitis, and H. Vollmer, editors. *Complexity of Constraints.* Springer Verlag, Berlin Heidelberg, 2007.

[Coo71] S. A. Cook. The complexity of theorem proving procedures. In *Proceedings 3rd Symposium on Theory of Computing*, pages 151–158. ACM Press, 1971.

[Dal97] V. Dalmau. Some dichotomy theorems on constant-free quantified boolean formulas. Technical Report LSI-97-43-R, Department de Llenguatges i Sistemes Informàtica, Universitat Politécnica de Catalunya, 1997.

[Dal00] V. Dalmau. *Computational Complexity of Problems over Generalized Formulas.* PhD thesis, Department de Llenguatges i Sistemes Informàtica, Universitat Politécnica de Catalunya, 2000.

[DHK05] A. Durand, M. Hermann, and P. G. Kolaitis. Subtractive reductions and complete problems for counting complexity classes. *Theoretical Computer Science*, 340(3):496–513, 2005.

[Hem04] E. Hemaspaandra. Dichotomy theorems for alternation-bounded quantified boolean formulas. *CoRR*, cs.CC/0406006, 2004.

[HV95] L. Hemaspaandra and H. Vollmer. The satanic notations: counting classes beyond #P and other definitional adventures. *Complexity Theory Column 8, ACM-SIGACT News*, 26(1):2–13, 1995.

[JCG97] P. Jeavons, D. Cohen, and M. Gyssens. Closure properties of constraints. *Journal of the ACM*, 44(4):527–548, 1997.

[KL99] H. Kleine Büning and T. Lettmann. *Propositional Logic: Deduction and Algorithms.* Cambridge Tracts in Theoretical Computer Science. Cambridge University Press, 1999.

[Lad89] R. E. Ladner. Polynomial space counting problems. *SIAM Journal on Computing*, 18(6):1087–1097, 1989.

[Lau06] D. Lau. *Function Algebras on Finite Sets.* Monographs in Mathematics. Springer Verlag, Berlin Heidelberg, 2006.

[Lev73] L. A. Levin. Universal sorting problems. *Problemi Peredachi Informatsii*, 9(3):115–116, 1973. English translation: *Problems of Information Transmission*, 9(3):265–266.

[MS72] A. R. Meyer and L. J. Stockmeyer. The equivalence problem for regular expressions with squaring requires exponential time. In *Proceedings 13th Symposium on Switching and Automata Theory*, pages 125–129. IEEE Computer Society Press, 1972.

[Pap94] C. H. Papadimitriou. *Computational Complexity.* Addison-Wesley, Reading, MA, 1994.

[Pip97] N. Pippenger. *Theories of Computability.* Cambridge University Press, Cambridge, 1997.

[PK79] R. Pöschel and L. A. Kalužnin. *Funktionen- und Relationenalgebren.* Deutscher Verlag der Wissenschaften, Berlin, 1979.

[Pös01] R. Pöschel. Galois connection for operations and relations. Technical Report MATH-LA-8-2001, Technische Universität Dresden, 2001.

[Rei05] O. Reingold. Undirected ST-connectivity in log-space. In *Proceedings 37th Symposium on Theory of Computing*, pages 376–385. ACM, 2005.

[Sch78] T. J. Schaefer. The complexity of satisfiability problems. In *Proceedings 10th Symposium on Theory of Computing*, pages 216–226. ACM Press, 1978.

[SM73] L. J. Stockmeyer and A. R. Meyer. Word problems requiring exponential time. In *Proceedings 5th ACM Symposium on the Theory of Computing*, pages 1–9. ACM Press, 1973.

[Sto77] L. Stockmeyer. The polynomial-time hierarchy. *Theoretical Computer Science*, 3:1–22, 1977.

[Tod91a] S. Toda. *Computational Complexity of Counting Complexity Classes.* PhD thesis, Tokyo Institute of Technology, Department of Computer Science, Tokyo, 1991.

[Tod91b] S. Toda. PP is as hard as the polynomial time hierarchy. *SIAM Journal on Computing*, 20:865–877, 1991.

[TW92] S. Toda and O. Watanabe. Polynomial time 1-Turing reductions from #PH to #P. *Theoretical Computer Science*, 100:205–221, 1992.

[Val79a] L. G. Valiant. The complexity of computing the permanent. *Theoretical Computer Science*, 8:189–201, 1979.

[Val79b] L. G. Valiant. The complexity of enumeration and reliability problems. *SIAM Journal of Computing*, 8(3):411–421, 1979.

[Vol94] H. Vollmer. *Komplexitätsklassen von Funktionen.* PhD thesis, Universität Würzburg, Institut für Informatik, Germany, 1994.

[Wra77] C. Wrathall. Complete sets and the polynomial-time hierarchy. *Theoretical Computer Science*, 3:23–33, 1977.

[Zan91] V. Zankó. #P-completeness via many-one reductions. *International Journal of Foundations of Computer Science*, 2:77–82, 1991.