The Complexity of Quantified Constraint Satisfaction Problems under Structural Restrictions *

Georg Gottlob
Inst. für Informationssysteme
Technische Universität Wien
A-1040 Vienna, Austria
gottlob@dbai.tuwien.ac.at

Gianluigi Greco
Dip. di Matematica
Università della Calabria
I-87030 Rende, Italy
ggreco@mat.unical.it

Francesco Scarcello
DEIS
Università della Calabria
I-87030 Rende, Italy
scarcello@deis.unical.it

Abstract

We give a clear picture of the tractability/intractability frontier for quantified constraint satisfaction problems (QCSPs) under structural restrictions. On the negative side, we prove that checking QCSP satisfiability remains PSPACE-hard for all known structural properties more general than bounded treewidth and for the incomparable hypergraph acyclicity. Moreover, if the domain is not fixed, the problem is PSPACE-hard even for tree-shaped constraint scopes. On the positive side, we identify relevant tractable classes, including QCSPs with prefix $\exists\forall$ having bounded hypertree width, and QCSPs with a bounded number of guards. The latter are solvable in polynomial time without any bound on domains or quantifier alternations.

1 Introduction

Quantified constraint satisfaction problems (QCSPs) are a generalization of constraint satisfaction problems (CSPs), where variables may be existentially and universally quantified, and nested quantifications are allowed. This framework is clearly much more expressive than plain existential-CSP, and may be fruitfully exploited for modeling a wide spectrum of problems from several domains.

A QCSP *instance* (or quantified constraint formula) ϕ is an expression of the form $Q_1\bar{V}_1 \cdots Q_m\bar{V}_m\ I$, where I is a constraint network (denoted by $CN(\phi)$), Q_i is a quantifier in $\{\exists, \forall\}$ (with $Q_i \neq Q_{i+1}$), and $\bar{V}_i$ is a set of variables, for $1 \leq i \leq m$. The string of quantifiers $Q_1 \cdots Q_m$ is called the *prefix* of ϕ. Recall that a *constraint network* is a triple $I = (Var, \bar{\mathcal{U}}, \mathcal{C})$, where Var is a finite set of variables, $\mathcal{U}$ is the set of domains $U(V)$, for each variable $V \in Var$, and $\mathcal{C} = \{C_1, C_2, \ldots, C_q\}$ is a finite set of constraints. A constraint $C_i = (S_i, r_i)$ consists of a list of variables S_i called *constraint scope*, and of a relation r_i, called *constraint relation*, providing C_i's allowed combinations of values for the variables in its scope. Sometimes it is more comfortable to denote C_i by its so called constraint atom $r_i(S_i)$. Then, the network I may be represented by the conjunction of all its constraint atoms. For simplicity, we limit our attention here to *closed* quantified constraint formulas, where all variables occurring in I are quantified. However, all our results may be easily extended to formulas with free variables.

As an example, consider the following quantified constraint formula ϕ_e: $\forall S, X, Y, T, R, U, P\ \exists V, Z\ \forall W\ a(S, X, T, R) \wedge b(S, Y, U, P) \wedge c(T, U, Z) \wedge d(W, X, Z) \wedge e(Y, Z) \wedge f(R, P, V) \wedge g(X, Y)$. This formula is a QCSP instance, whose constraint network $CN(\phi_e)$ is represented by the constraint atoms occurring in the conjunction. The quantifier prefix (short: prefix) of ϕ_e is the string $\forall\exists\forall$.

Not surprisingly, the increased expressive power of QCSPs comes at a cost. Indeed, while deciding the satisfiability of traditional (i.e., purely existential) CSPs is NP-complete, this problem is PSPACE-complete [Borner *et al.*, 2003], in the general quantified setting. Hence, much effort has been spent to identify *tractable classes* of QCSPs.

These approaches can be divided into two main groups: techniques that identify tractable classes of QCSPs by exploiting particular properties of constraint relations, and techniques that identify tractable classes by exploiting the structure of constraint scopes, usually known as *structural decomposition methods*. While several deep results have been already achieved by techniques in the former group (see, e.g., [Borner *et al.*, 2003; Bulatov *et al.*, 2000; Bunind *et al.*, 1995; Chen, 2004a; Creignou *et al.*, 2001; Jeavons *et al.*, 1997]), only a few papers focused on structural decomposition methods, though they were proven to be useful in the non-quantified setting (see, e.g., [Dechter, 2003; Gottlob *et al.*, 2000]).

Recall that the structure of constraint network I is best represented by its associated hypergraph $\mathcal{H}(I) = (V, H)$, where $V = Var$ and $H = \{var(S) \mid C = (S, r) \in \mathcal{C}\}$, and $var(S)$ denotes the set of variables in the scope S of constraint C. Some graph-based techniques are based on the *primal graph* $G(\mathcal{H}(I)) = (V, E)$ of $\mathcal{H}(I)$, where two variables are connected in E if they occur together in some hyperedge (i.e., in the scope of some constraint).

Chen recently presented an interesting result about structurally tractable QCSPs [Chen, 2004b]. He describes a polynomial-time algorithm for classes of QCSPs having (primal graphs with) bounded treewidth, fixed domain, and fixed prefix. In fact, the complexity of this algorithm depends

*This work was supported by the Austrian Science Fund (FWF) project: *Nr. P17222-N04 Complementary Approaches to Constraint Satisfaction*.

dramatically on the number of quantifier alternations and on the size of the largest variable domain. As noted in [Chen, 2004b], the same result has been independently derived by [Feder and Kolaitis, 2004], by exploiting Courcelle's theorem about monadic second order logic on bounded treewidth structures.

Notice that there is no indication that these results are optimal, and in fact several interesting questions arose, and will be the subject of this paper:

(1) Are QCSPs having bounded treewidth tractable if domains are not fixed?

(2) May we extend this result to other structural notions, possibly more general than bounded treewidth?

(3) Are there different kind of restrictions on quantified constraint formulas that make QCSPs tractable?

The answers to these questions comprise both good news and bad news. We prove strong hardness results, but we also identify new tractable classes of QCSPs, having neither fixed bound on domains nor fixed bound on quantifier alternations. Our main contributions, shown in Figure 1, are the following:

- ► We prove that, without the fixed domain restriction, even QCSP instances whose structure is a tree and whose prefix is $\forall\exists$ are co-NP-hard. Moreover, adding further alternations we get complete problems for all levels of the polynomial hierarchy. It follows that this problem is PSPACE-complete if there are no bounds on the quantifier prefix.
- ► On the positive side, we prove that, if the prefix is $\exists\forall$ (or some substring of it), then solving acyclic QCSPs is feasible in LOGCFL and hence in polynomial time. Moreover, this tractability extends to all known generalizations of acyclicity, and in particular to bounded hypertree-width QCSPs [Gottlob *et al.*, 2000].
- ► We prove that, for fixed domains, the tractability result for bounded treewidth is almost optimal. Indeed, solving QCSPs over the binary domain $\{0, 1\}$ remains PSPACE-complete even if the structure is an acyclic hypergraph, whose incidence graph has bounded treewidth, and whose primal graph has small (i.e., logarithmic) treewidth.
- ► All these results show that traditional structural techniques do not help very much, but for some simple cases and with limited quantification. Indeed, our hardness proofs show that the presence of quantifiers radically alters the structural properties of the constraint scopes. We thus realize that it is worthwhile taking into account how they interact with the scope structure, and in fact considering quantifiers as part of the scope structure itself. Following this idea, we identify a different kind of restriction on quantified constraint formulas that ensure tractability and that is incomparable with the other structural classes. In particular, for any fixed k, we define the class k-GQCSP of k-guarded QCSPs, that are solvable in polynomial time, without any restriction on domains or quantifier alternations.

2 Quantified CSPs

Let $I = (Var, \mathcal{U}, \mathcal{C})$ be a constraint network. An assignment σ for a set of variables $\bar{V} \subseteq Var$ is a function mapping each variable $V \in \bar{V}$ onto its domain $U(V) \in \mathcal{U}$. If $\bar{V} = Var$, σ is said *complete*, otherwise it is a partial assignment. We say that a complete assignment σ satisfies I, denoted by $\sigma \models I$, if for each constraint $(S_i, r_i) \in \mathcal{C}$, $\sigma(S_i) \in r_i$. An extension of σ to a set $\bar{V}' \supset \bar{V}$ is an assignment σ' for $\bar{V}'$ such that $\sigma'(V) = \sigma(V)$ for each $V \in \bar{V}$. We denote by $ext(\sigma, \bar{V}')$ the set of all the extensions of σ to $\bar{V}'$. For the trivial assignment $\sigma_\emptyset$ for the empty set of variables, $ext(\sigma_\emptyset, \bar{V}')$ is clearly the set of all assignments for $\bar{V}'$.

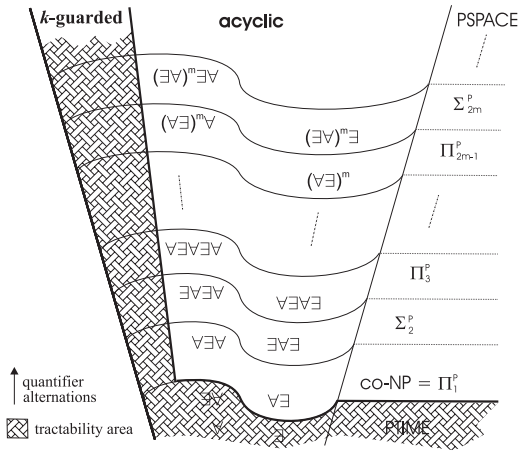

Figure 1: Structural restrictions and (in)tractable QCSPs.

Let $\phi : Q_1\bar{V}_1Q_2\bar{V}_2Q_3\bar{V}_3 \ldots Q_m\bar{V}_m\ I$ be a QCSP instance, and let σ_0 be the trivial assignment $\sigma_\emptyset$. A *strategy* for ϕ is any function s such that, for each pair Q_i, σ_{i-1}, with $1 \leq i \leq m$, $s(Q_i, \sigma_{i-1})$ is either one assignment in $ext(\sigma_{i-1}, \bar{V}_i)$, if $Q_i = \exists$, or the whole set of possible extensions $ext(\sigma_{i-1}, \bar{V}_i)$, if $Q_i = \forall$. A complete assignment σ_m is *derivable* from a strategy s if there are $m-1$ assignments $\sigma_1, \ldots, \sigma_{m-1}$ such that $\sigma_i \in s(Q_i, \sigma_{i-1})$, for any $1 \leq i \leq m$. Then, s is a *solution* for ϕ if all derivable assignments satisfy I. A QCSP instance is *satisfiable* iff it has a solution.

It is worthwhile noting that, in the definition of QCSPs, different variables have different domains, in general. This is especially useful in the quantified setting. However, in the literature, QCSPs are sometimes defined over a unique domain U or, equivalently, with the same domain $U(V)$ for each variable V. We say that such QCSPs are *untyped*, in contrast to the general ones, called *typed*. The following proposition shows that the two formalisms are in fact logically equivalent.

Proposition 2.1 *For any QCSP instance ϕ, there exists an untyped equivalent instance ϕ'. Moreover, if $CN(\phi)$ is a binary network, ϕ' can be computed in polynomial time.*

Notice that going from typed to untyped instances may be exponential for non-binary networks, as the former setting allows more succinct and efficient representations.

We remark that all complexity results in this paper hold for both settings. Indeed, we prove membership results and provide algorithms for the general typed setting, and prove hardness results by using either binary networks, or a unique binary domain for all variables.

We say that a class S of hypergraphs has the bounded hypertree width property, denoted by BHTW, if there is a $k > 0$ such that every hypergraph in S has hypertree width at most k [Gottlob *et al.*, 2000]. Similarly, we define the property BTW, meaning bounded treewidth [Robertson and Seymous, 1986] of the primal graph (of the constraint hypergraph), and the property BITW, meaning bounded treewidth of the inci-

dence graph. Moreover, we say that a class S of hypergraphs has the small hypertree width property, denoted by SHTW, if the hypertree width of every hypergraph $\mathcal{H} \in S$ is at most $\log |\mathcal{H}|$. The small treewidth property STW of primal graphs is defined similarly. We also consider the property ACYCLIC (resp., TREES) of any class of acyclic hypergraphs (resp., primal graphs).

We study how the complexity of QCSPs change as a function of quantifier alternations and of constraint structures. Moreover, we distinguish the case of arbitrary domains, denoted by ANY, and of binary domains, denoted by $\{0, 1\}$.

Let $\bar{Q}$ be a string of quantifier alternations, $\mathcal{S}$ a hypergraphs property, and $\mathcal{D}$ a domain property in $\{\text{ANY}, \{0, 1\}\}$. Then, QCSP$(\bar{Q}, \mathcal{S}, \mathcal{D})$ is the problem of deciding whether an instance $\phi \in class(\bar{Q}, \mathcal{S}, \mathcal{D})$ is satisfiable, where $class(\bar{Q}, \mathcal{S}, \mathcal{D})$ is any class of QCSP instances over domains of kind $\mathcal{D}$, with alternation prefix $\bar{Q}$, and whose associated hypergraphs have property $\mathcal{S}$.

3 Structural methods do not help very much

3.1 Some tractable instances

From [Gottlob *et al.*, 2000], we already know that QCSP$(\exists, \text{BHTW}, \text{ANY})$ is in polynomial time, and the same holds for any structural restriction stronger than bounded hypertree-width. Moreover, it is easy to see that QCSP$(\forall, \text{BHTW}, \text{ANY})$ is even easier. We next show that also QCSP$(\exists\forall, \text{BHTW}, \text{ANY})$ is tractable.

Let ϕ be a QCSP instance, $I = (Var, \mathcal{U}, \mathcal{C})$ the constraint network of ϕ, and $\bar{Y}$ be a set of universally quantified variables. Let $r(\bar{X}, \bar{Y}')$ a constraint of I over variable sets $\bar{X}$ and $\bar{Y}'$, where $\bar{Y}'$ is the set of $\bar{Y}$ variables occurring in the scope of r. Denote by $cart(\bar{Y}')$ the relation containing all combination of values from the domains of variables in $\bar{Y}'$, i.e., $cart(\bar{Y}') = \times_{Y \in \bar{Y}'} U(Y)$.

Then, define $\bar{Y}$-$red(r)$ as the relation containing all and only those tuples t of relation r such that, for any combination t' of values for variables $\bar{Y}'$, $t[\bar{X}] \cdot t' \in r$. Note that, for the special case $\bar{Y}' = \emptyset$, we get $\bar{Y}$-$red(r) = r$; for the special case $\bar{X}' = \emptyset$, we simply require all combinations of values for $\bar{Y}'$, that is, $\bar{Y}$-$red(r) = r$, if r is precisely $cart(\bar{Y}')$, otherwise it is the empty relation.

Denote by $\bar{Y}$-$red(\phi)$ the QCSP obtained from ϕ by replacing each constraint relation r by $\bar{Y}$-$red(r)$.

Lemma 3.1 *For any QCSP ϕ and set of universally quantified variables $\bar{Y}$, $\bar{Y}$-$red(\phi)$ can be computed in logspace.*

After the above lemma and exploiting the fact that no useful assignment is lost with this transformation, we can show the following.

Theorem 3.2 QCSP$(\exists\forall, \text{BHTW}, \text{ANY})$ *is in polynomial time. Moreover, it is* LOGCFL*-complete, and hence tractable and parallelizable.*

3.2 Encoding Boolean formulas as acyclic QCSPs

To any CNF formula $\Phi = c_1 \wedge \ldots \wedge c_m$ over Boolean variables $\bar{V} = \{V_1, \ldots, V_n\}$, we *associate* a binary acyclic constraint network $I(\Phi) = (Var, \mathcal{U}, \mathcal{C})$. This constraint network will be used hereafter for characterizing the complexity of acyclic and quasi-acyclic quantified CSPs.

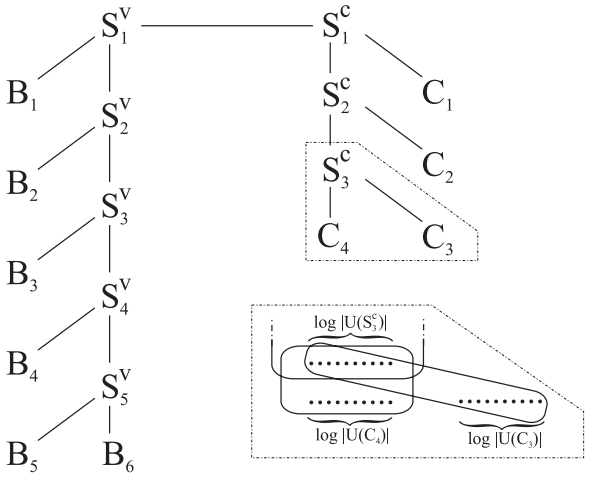

Figure 2: Constraint network $I(\bar{\Phi})$ in Theorem 3.4. In the right-bottom box, encodings of constraints in Theorem 3.7.

Consider the following CNF formula, that we use as a running example: $\bar{\Phi} = (V_1 \vee V_2 \vee V_3) \wedge (V_1 \vee \neg V_4) \wedge (V_1 \vee V_6 \vee V_4) \wedge (V_2 \vee V_6)$. Then, Figure 2 shows the constraint structure (i.e. the constraint hypergraph) of $I(\bar{\Phi})$. Note that in this case hypergraph and primal graph representations coincide, as $CN(\bar{\Phi})$ contains only binary constraints.

The set of variables Var is the union of a set of *clause variables* $\bar{C} = \{C_j \mid 1 \leq j \leq m\}$ corresponding to the m clauses of Φ, of a set $\bar{B} = \{B_i \mid V_i \in \bar{V}\}$, corresponding to the n Boolean variables occurring in Φ, and of two sets $\bar{S^c} = \{S^c_j, \mid 1 \leq j \leq m-1\}$ and $\bar{S^v} = \{S^v_i, \mid 1 \leq i \leq n-1\}$ of special variables, called *clause selectors* and *variable selectors*, respectively.

For a variable V_i of Φ, the domain of $I(\Phi)$ contains literal constants v_i and $\neg v_i$ associated with its truth-values. Moreover, for any clause c_j in which V_i occurs, the domain of $I(\Phi)$ contains a literal constant in $\{v^j_i, \neg v^j_i\}$ encoding the truth value for V_i that makes c_j true. We denote by l_i (resp., l^j_i) any of these (positive or negative) literals, and by $\neg l_i$ (resp., $\neg l^j_i$) its complement. Moreover, we denote by $satLit(c_j)$ the set of literals that make c_j true. E.g., for $c_2 = (V_1 \vee \neg V_4)$ in $\bar{\Phi}$, $satLit(c_2) = \{v^2_1, \neg v^2_4\}$.

In more detail, the domains of $I(\Phi)$ variables are the following: for any variable C_j (corresponding to clause c_j of Φ), $U(C_j) = satLit(c_j)$; for any Boolean variable B_i, $U(B_i) = \{v_i, \neg v_i\}$; for any clause-selector S^c_j, $U(S^c_j) = \bigcup_{j' \geq j} U(C_{j'})$; finally, for any variable-selector S^v_i, $U(S^v_i) = \bigcup_{i' \geq i} U(B_{i'})$.

Intuitively, Boolean variables encode a truth-value assignment to Φ, whereas any clause variable C_j chooses some literal in $satLit(c_j)$ that satisfies it. Any selector variable may take a value coming from either variable connected to it. Thus, any choice of all variable selectors corresponds to the propagation of a literal value l_i coming from some Boolean variable B_i in the left branch. Similarly, any choice of clause selectors corresponds to the propagation in the right branch of some literal l^j_k satisfying some clause c_j. If for all possible propagations from both branches, no pair $l_i, \neg l^j_i$ of comple-

mentary literals meets at the topmost constraint of the network, then the values of Boolean variables encode a satisfying truth-value assignment for Φ. We next describe the constraints in $\mathcal{C}$ that implement the above idea, where the indices i and i' actually are values in the interval $[1 \dots n]$ and where the indices j, j' are values in $[1 \dots m]$.

- The topmost constraint (S_1^v, S_1^c), called $evaluate$, has a constraint relation consisting of $\{(l_i, \neg l_i^j) \mid l_i^j \in satLit(c_j)\}$. Note that $evaluate$ is satisfied only by assignments where its variables take complementary literals.
- For any constraint (S_i^v, B_i) between a variable-selector S_i^v and a Boolean variable B_i, its constraint relation consists of the tuples: $\{(l_i, l_i)\} \cup \{(l_{i'}, l_i) \mid i' > i\}$.
- For any constraint between a pair of adjacent variable-selectors (S_i^v, S_{i+1}^v) and for the constraint (S_i^v, B_n) with $i = n-1$, the constraint relation is $\{(l_{i'}, l_{i'}) \mid i' > i\} \cup \{(l_i, l_{i'}) \mid i' > i\}$.
- For any constraint (S_j^c, C_j) between a clause-selector S_j^c and a clause variable C_j, its constraint relation consists of the tuples: $\{(l_h^j, l_h^j) \mid l_h^j \in satLit(c_j)\} \cup \{(l_k^{j'}, l_h^j) \mid l_h^j \in satLit(c_j), l_k^{j'} \in satLit(c_{j'}), j' > j\}$.
- For any constraint between a pair of adjacent clause-selectors (S_j^c, S_{j+1}^c) and for the constraint (S_j^c, C_m) with $j = m-1$, the constraint relation consists of the tuples: $\{(l_h^{j'}, l_h^{j'}) \mid l_h^{j'} \in satLit(c_{j'}), j' > j\} \cup \{(l_k^j, l_h^{j'}) \mid l_k^j \in satLit(c_j), l_h^{j'} \in satLit(c_{j'}), j' > j\}$.

If μ is a truth-value assignment for all variables $\bar{V}$ of Φ, then, σ_μ, denotes the assignment such that $\sigma_\mu(B_i) = v_i$ if $\mu(V_i) = true$, and $\sigma_\mu(B_i) = \neg v_i$ if $\mu(V_i) = false$.

Lemma 3.3 *Let μ be a truth-value assignment for all variables $\bar{V}$ of Φ and σ_μ the corresponding assignment for $\bar{B}$. Then, μ is a satisfying assignment for Φ if and only if there exists an assignment $\sigma' \in ext(\sigma_\mu, \bar{C})$ such that, for every $\sigma'' \in ext(\sigma', \bar{S}^v \cup \bar{S}^c)$, $\sigma'' \not\models I(\Phi)$ holds.*

After this lemma, we immediately get that, even for acyclic binary constraint networks with just two quantifier alternations, solving a quantified CSP is intractable.

Theorem 3.4 $\mathrm{QCSP}(\forall\exists, \mathtt{TREES}, \mathtt{ANY})$ *is* co-NP*-hard.*

Proof. From a CNF Boolean formula Φ, we build in polynomial time $QC(\Phi) = \forall\bar{B}, \bar{C}\ \exists\bar{S}\ I(\Phi)$, where $I(\Phi)$ is the acyclic constraint network associated with Φ. From Lemma 3.3, satisfying assignments for Φ are in one-to-one correspondence with assignments to the variables in $\bar{B}$ and $\bar{C}$ such that all their complete extensions do not satisfy $I(\Phi)$. Thus, Φ is not satisfiable iff $QC(\Phi)$ is satisfiable. □

3.3 Intractable acyclic instances

After having shown in the previous section the tractability for $\exists\forall$ and the intractability for $\forall\exists$, we now settle the complexity of acyclic QCSPs with arbitrary quantifier prefixes.

Theorem 3.5 *For any natural number $m \geq 1$,*

1. $\mathrm{QCSP}((\forall\exists)^m\forall, \mathtt{BHTW}, \mathtt{ANY})$ *is Π_{2m-1}^P-complete, and hardness holds for* $\mathrm{QCSP}((\forall\exists)^m, \mathtt{TREES}, \mathtt{ANY})$*, too;*
2. $\mathrm{QCSP}((\exists\forall)^m\exists\forall, \mathtt{BHTW}, \mathtt{ANY})$ *is Σ_{2m}^P-complete, and hardness holds for* $\mathrm{QCSP}((\exists\forall)^m\exists, \mathtt{TREES}, \mathtt{ANY})$*, too.*

Proof. For space limitations we only prove *Hardness* for Point 2, here. For any $m \geq 1$, consider the Σ_{2m}^P-complete problem of deciding whether a quantified Boolean formula $\Psi' = \exists V_1 \forall V_2 \cdots \forall V_{2m} \neg\Phi$ is satisfiable, where Φ is in CNF. From this formula, we build in polynomial time the following instance of $\mathrm{QCSP}((\exists\forall)^m\exists, \mathtt{TREES}, \mathtt{ANY})$: $QC(\Psi') = \exists B_1 \forall B_2 \cdots \forall B_{2m}, \bar{C}\ \exists\bar{S}\ I(\Phi)$. From Lemma 3.3, it can be seen easily that Ψ' is satisfiable iff $QC(\Psi')$ is satisfiable.

Membership. The proof is by induction. First observe that, given any instance $Q = \exists\bar{V}_1\forall\bar{V}_2\phi$ of $\mathrm{QCSP}(\exists\forall, \mathtt{BHTW}, \mathtt{ANY})$, its complementary problem Q^c (deciding whether for all assignment σ to $\bar{V}_1$ there exists an extension to $\bar{V}_2$ that does not satisfy ϕ) is in LOGCFL and hence in polynomial time, by Theorem 3.2 and the fact that LOGCFL is closed under complementation. Then, we prove the basis of the induction, $m = 1$. The problem $\mathrm{QCSP}(\forall\exists\forall, \mathtt{BHTW}, \mathtt{ANY})$ is in Π_1^P = co-NP. Indeed, let $Q = \forall\bar{V}_1\exists\bar{V}_2\forall\bar{V}_3\phi$ be any instance of this problem. Then, its complement can be decided in NP: guess an assignment σ to $\bar{V}_1$ and check that for all assignment $\sigma' \in ext(\sigma, \bar{V}_2)$ there is a complete assignment $\sigma'' \in ext(\sigma', \bar{V}_3)$ that does not satisfy all the constraints in ϕ. From the observation above, this check is feasible in polynomial time. Moreover, $\mathrm{QCSP}(\exists\forall\exists\forall, \mathtt{BHTW}, \mathtt{ANY})$ is in Σ_2^P, as any instance of this problem may be solved by a non-deterministic Turing machine with an oracle for $\mathrm{QCSP}(\forall\exists\forall, \mathtt{BHTW}, \mathtt{ANY})$. The induction step is a simple adaptation of the above reasoning for any $m > 1$. □

Corollary 3.6 *The quantified constraint satisfaction problem (on arbitrary domain) is* PSPACE*-complete, even if restricted on constraint networks whose structure is a tree.*

3.4 Fixed domain helps only with fixed arity

We now show that hypergraph acyclicity does not help in making easy the QCSP problem, even if we consider Boolean domains only. The same holds even in case we additionally require that the incidence graph has bounded treewidth, and the primal graph has small (logarithmic) treewidth. This entails that the problem remains intractable as long as we have non-fixed arities, even for very simple constraint interactions.

Theorem 3.7 *For any natural number $m \geq 1$,*

- $\mathrm{QCSP}((\forall\exists)^m, \mathtt{ACYCLIC} \cap \mathtt{BITW} \cap \mathtt{STW}, \{\mathtt{0}, \mathtt{1}\})$ *is Π_{2m-1}^P-complete;*
- $\mathrm{QCSP}((\exists\forall)^m\exists, \mathtt{ACYCLIC} \cap \mathtt{BITW} \cap \mathtt{STW}, \{\mathtt{0}, \mathtt{1}\})$ *is Σ_{2m}^P-complete.*

Proof. Let Φ be a Boolean formula, and $I(\Phi) = (Var, \mathcal{U}, \mathcal{C})$ its associated acyclic constraint network. We consider the network $I'(\Phi) = (Var', \{0,1\}, \mathcal{C}')$, defined as follows. For each variable $X \in Var$, Var' contains $|U(X)|$ distinct variables $X_1, ..., X_{\log|U(X)|}$, with domain $U'(X_i) = \{0,1\}$ for each X_i. For each constraint $r(X,Y)$ in $\mathcal{C}$, $\mathcal{C}'$ contains a constraint $r'(X_1, ..., X_{\log|U(X)|}, Y_1, ..., Y_{\log\ |U(Y)|})$, whose constraint relation is such that, for each tuple (x_i, y_j) in r, r' contains the tuple $(enc(x_i), enc(y_j))$, where the string of bits $enc(x_i)$ (resp. $enc(y_j)$) is the binary encoding of domain value x_i (resp. x_j). The right-bottom box in Figure 2 shows a portion of the constraint network $I'(\bar{\Phi})$ associated to the formula $\bar{\Phi}$ of our running example. Observe that the constraint network $I'(\Phi)$ is in $\mathtt{ACYCLIC} \cap \mathtt{BITW} \cap \mathtt{STW}$. Indeed,

the hypergraph associated to $I'(\Phi)$ is acyclic, and the number of variables in each hyperedge is bounded by $2 \log c$, where c is the size of largest domain over all the variables in Var. Therefore, the treewidth of the primal graph is at most $2 \log c$. Moreover, it is easy to check that the treewidth of the incidence graph of $I'(\Phi)$ is 3. Finally, observe that there exists a one-to-one correspondence between assignments to variables in $I(\Phi)$ and in $I'(\Phi)$, and thus the result immediately follows from Theorem 3.5. □

Corollary 3.8 *The quantified constraint satisfaction problem is* PSPACE*-complete, even if restricted on Boolean constraint networks whose structure is in* ACYCLIC ∩ BITW ∩ STW.

4 Guarded formulas and tractable CSPs

In this section, we describe a wide class of quantified constraint formulas that are tractable, even if there is no constant bound on domain sizes or quantifier alternations. Recall that any QCSP instance ϕ may be represented by a logical expression, as shown in Section 1 for QCSP ϕ_e. Technically, let us denote the pure logical formula of ϕ (without the encoding of relations, domains, etc.) by $form(\phi)$.

Following [Kolaitis *et al.*, 2000], we denote by $\mathrm{FO}_{\wedge,+}$ the fragment of first order sentences where arbitrary quantifications and conjunctions are allowed, but where negations and disjunctions are forbidden. They observed that the existential fragment $\exists\mathrm{FO}_{\wedge,+}$ of $\mathrm{FO}_{\wedge,+}$ has the same expressive power as the constraint satisfaction problems. By allowing any kind of quantifiers, this observation may be clearly extended to the connection between general $\mathrm{FO}_{\wedge,+}$ formulas and quantified constraint formulas. Notice that, in this more general setting, there are different equivalent logical representation of the same instance. For example, one may use parentheses for distinguishing subformulas and delimiting quantifier scopes. In this section, we represent QCSPs by $\mathrm{FO}_{\wedge,+}$ formulas that are not necessarily in the traditional prenex form. Notice, however, that each $\mathrm{FO}_{\wedge,+}$ formula can be easily transformed into an equivalent prenex formula.

4.1 The fragment k-GQCSP of k-guarded QCSPs

We show that, for each constant k, a simple and appealing fragment of $\mathrm{FO}_{\wedge,+}$ is decidable in polynomial time. Since we have no bound on the number of variables in a formula, we assume w.l.o.g. that each variable is quantified over only once, i.e., quantified variables are not reused. In the following, we denote by $free(\psi)$ the free variables of a logical formula ψ.

Definition 4.1 The class k-GQCSP of k-guarded QCSPs consists of those QCSPs instances ϕ whose formula $form(\phi)$ belongs to the fragment G_k^* of $\mathrm{FO}_{\wedge,+}$ defined as follows. G_k^* is the smallest subset of $\mathrm{FO}_{\wedge,+}$ such that:

- every atom belongs to G_k^*;
- if ϕ_1 and $\phi_2 \in G_k^*$, then $\phi_1 \wedge \phi_2 \in G_k^*$;
- let $\alpha_1, \ldots, \alpha_i$ be atoms, where $i \leq k$, and let ψ be a formula in G_k^*. If the free variables $free(\psi) \subseteq var(\alpha_1) \cup \cdots \cup var(\alpha_i)$, then, for each tuple of variables $\bar{y}$ and each quantifier $Q \in \{\exists, \forall\}$, the formula $\psi' : \ Q\bar{y}(\alpha_1 \wedge \cdots \wedge \alpha_i \wedge \psi)$ belongs to G_k^*. The set of atoms $\{\alpha_1, \ldots, \alpha_i\}$ is referred to as the guard of ψ' and is denoted by $guard(\psi')$. □

Example 4.2 Consider again QCSP instance ϕ_e presented in the Introduction, and the following equivalent instance, where $form(\phi_e)$ is rewritten as

$$\psi = \forall S, X, Y, T, R, U, P(\ a(S, X, T, R) \ \wedge \ b(S, Y, U, P) \wedge \exists V\ f(R, P, V) \ \wedge \ \exists Z\ (\ g(X, Y) \ \wedge \ \ c(T, U, Z) \wedge \forall W\ d(W, X, Z) \ \wedge \ e(Y, Z)\ )\ )$$

This is a 2-guarded constraint formula, i.e., $\psi \in G_2^*$. The guard of formula ψ is $guard(\psi) = \{a(S, X, T, R), b(S, Y, U, P)\}$. For the formulas $\psi_1 = \exists V\ \ f(R, P, V)$, $\psi_2 = \exists Z\ (\ g(X, Y) \ \wedge \ c(T, U, Z) \ \wedge \ \forall W\ d(W, X, Z) \ \wedge \ e(Y, Z)\ )$ and $\psi_3 = \forall W\ \ d(W, X, Z)$, we have the following guards: $guard(\psi_1) = \{f(R, P, V)\}$, $guard(\psi_2) = \{g(X, Y), c(T, U, Z)\}$, and $guard(\psi_3) = \{d(W, X, Z)\}$. □

Note that the above definition of k-guardedness is congenial to the specific syntax of quantified CSPs and differs from that of k-guarded first order logic [Andreka *et al.*, 1998; Grädel, 1999]. In the standard formalisms of the guarded fragment GF of first order logic or of the k-guarded fragment GF_k of first order logic [Gottlob *et al.*, 2003], the guards of an existentially quantified subformula ψ are added conjunctively to ψ ($guard(\psi) \wedge \psi$), just as for G_k^*. However, the guards of a universal formula ψ are added *in form of an implication*: $guard(\psi) \rightarrow \psi$. This is much more natural, since these logics have negation and guardedness needs to be correctly preserved under negations. For example, the negation of a guarded formula $\neg\exists\bar{Y}(g(\bar{Y}) \wedge \psi(\bar{Y}))$ is logically equivalent to $\forall\bar{Y}(g(\bar{Y}) \rightarrow \neg\psi(\bar{Y}))$. Since the logic $\mathrm{FO}_{\wedge,+}$ of constraint formulas has conjunction ($\wedge$) as unique binary connective, it is syntactically impossible to express an implication $guard(\psi) \rightarrow \psi$ in $\mathrm{FO}_{\wedge,+}$ (in fact, this is also semantically impossible). On the other hand, since negation is missing in $\mathrm{FO}_{\wedge,+}$, no problems involving "wrong guards" can arise through negation when using the natural (but non-standard) guards introduced for the above defined fragments G_k^* of $\mathrm{FO}_{\wedge,+}$.

For the k-guarded fragment GF_k of first order logic (i.e., for the standard k-guarded fragment) the following tractability result was shown in [Gottlob *et al.*, 2003]:

Proposition 4.3 *The combined complexity of evaluating a GF_k formula ϕ over a set of finite relations D is in $O(|\phi| \times |D|^k)$.*

Let us now show that also the class k-GQCSP of k-guarded QCSPs is tractable, notwithstanding the "nonstandard" guards for universally quantified subformulas.

Lemma 4.4 *There is an algorithm TRANSFORM which for each QCSP $\phi \in k$-GQCSP computes a pair (D, ϕ^*) where D is a finite database and $\phi^* \in GF_k$, such that ϕ is satisfied iff $D \models \phi^*$. The TRANSFORM algorithm runs in logspace.*

Proof. Let $\phi \in k$-GQCSP, and let R be the set of constraint relations of $CN(\phi)$. Consider a subformula $\psi' = \forall\bar{Y}(guard(\psi) \wedge \psi)$ of $form(\phi)$, where $\forall\bar{Y}$ is a maximal prefix of universally quantified variables, and where $guard(\psi)$ is a conjunction of m atoms $r_i(\bar{Y}_i, \bar{X}_i)$, with $1 \leq i \leq m \leq k$, where $\bar{Y}_i$ are the variables in its scope included in $\bar{Y}$, and $\bar{X}_i$ its other variables, hence $\bar{X}_i \cap \bar{Y} = \emptyset$. For $1 \leq i \leq m$, we denote by $r_i\!\downarrow$ the relation obtained from the constraint relation r_i by keeping all $\bar{Y}_i$-columns and projecting out all $\bar{X}_i$ columns, i.e., $r_i\!\downarrow := \prod_{\bar{Y}_i} r_i$.

Assume that for some $1 \leq i \leq m$, $r_i\downarrow$ is not equal to the Cartesian product $cart(\bar{Y}_i)$ of $\bar{Y}_i$ domains. That is, $r_i\downarrow$ does not consist of precisely all possible combinations of constants for all $\bar{Y}$ variables occurring in r_i. Then one tuple $\bar{a}$ of such constants is not in $r_i\downarrow$ and thus the atom $r_i(\bar{Y}_i, \bar{X}_i)$ cannot be satisfied for the substitution $\bar{Y}_i = \bar{a}$. Since r_i occurs positively in ϕ, ϕ cannot be satisfied. The key observation is thus that, whenever ϕ is satisfied, for $1 \leq i \leq m$, $r_i\downarrow$ must be equal to $cart(\bar{Y}_i)$. Algorithm TRANSFORM starts by checking whether this is true. From Lemma 3.1, this check can be implemented to run in logspace: just compute $\bar{Y}$-$red(r_i\downarrow)$, and check that it is not empty. If this test fails for at least one guard relation r_i, then TRANSFORM outputs the formula *false*. Otherwise, let g_ψ be a new constraint atom with constraint scope $\bar{Y}$, whose relation is the join of the $r_i\downarrow$ relations. Now, consider the set of free variables $\bar{X} = free(\psi')$. If $\bar{X} \neq \emptyset$, since ϕ is a k-guarded formula, there are at most k atoms in ϕ that cover $\bar{X}$. Let $g_{\psi'}$ be a new constraint atom with constraint scope $\bar{X}$, whose relation is the projection over $\bar{X}$ of the join of these guard atoms. Moreover, let g be a new constraint atom with constraint scope $\bar{X} \cup \bar{Y}$, whose relation is the join of g_ψ and $g_{\psi'}$ (or just g_ψ, if $\bar{X}$ is empty). Note that, since k is fixed, TRANSFORM may compute g from its input ϕ in logspace. Then, TRANSFORM replaces our subformula $\psi' = \forall\bar{Y}(guard(\psi) \wedge \psi)$ by the equivalent subformula $\psi'' = \forall\bar{Y}(g \rightarrow guard(\psi) \wedge \psi)$. Note that this transformation does not change the set of free variables, as $free(\psi') = free(\psi'')$. Thus, the set of atoms that cover these variables and acts as the guard of ψ' in $form(\phi)$ will be the guard of ψ'' in the new formula.

After having done this for all universal guarded subformulas, TRANSFORM has generated a formula $\phi^* \in GF_k$. Let D the database consisting of all constraint relations of $CN(\phi)$ plus all relations like g computed for modifying the universal quantifications. It can be seen that ϕ is satisfiable iff $D \models \phi^*$. Moreover, the entire algorithm runs in logspace. □

Theorem 4.5 *For each fixed k, the satisfiability of any QCSP in the class k-*GQCSP *can be checked in polynomial time.*

Proof. Follows from Lemma 4.4 and Proposition 4.3. □

4.2 Extending the k-GQCSP fragment

When looking at the fragment k-GQCSP, and even more generally, at QCSPs in which universal quantifiers appear, we observe that the expressive power of universal quantification is rather poor. In fact, as already observed in the proof of Lemma 4.4, a subformula $\forall\bar{Y}r(\bar{Y}, \bar{X})$ can only be true if the projection over $\bar{Y}$ of r is equal to the cartesian product $cart(\bar{Y})$ representing the set of *all* tuples that can be composed from all possible domain elements from the respective domains. This is a rather stringent condition. On the other hand, by standard k-guards of the form $(r_1 \wedge r_2 \wedge \cdots \wedge r_m) \rightarrow \psi$, we can express some other interesting properties. For example by standard guards, one may express an *inclusion dependency* stating that part of one constraint relation must be contained in another constraint relation. To add expressive power, we thus suggest to allow the use of standard guards together with conjunctively specified guards (for universally quantified subformulas). We thus define the fragment k-GQCSP$^+$ just as k-GQCSP in Definition 4.1 with the following addition: *If the free variables $free(\psi) \subseteq var(\alpha_1) \cup \cdots \cup var(\alpha_i)$, then, for each tuple of variables $\bar{y}$ the formula $\psi' : \forall\bar{y}(((\alpha_1 \wedge \cdots \wedge \alpha_i) \rightarrow \psi)$ belongs to G_k^*.*

After the results in Section 4.1, it is easy to see that, for each fixed k, the satisfiability of any QCSP in the class k-GQCSP$^+$ can be checked in polynomial time. Moreover, we are able to show that k-GQCSP$^+$ is strictly more expressive than k-GQCSP (for space reasons, we defer the proof to the full paper).

References

[Andreka *et al.*, 1998] H. Andreka, J. van Benthem, and I. Nemeti. Modal languages and bounded fragments of predicate logic. *Journal of Philosophical Logic*, 27(3):217–274, 1998.

[Borner *et al.*, 2003] F. Borner, A. Bulatov, A. Krokhin, and P. Jeavons. Quantified Constraints: Algorithms and Complexity. *In Proc. of CSL'03*, pp. 58–70, 2003.

[Bulatov *et al.*, 2000] A. Bulatov, A. Krokhin, and P. Jeavons. Constraint satisfaction problems and finite algebras. *In Proc. of ICALP'00*, pp. 272–282.

[Bunind *et al.*, 1995] H.K. Buning, M. Karpinski, and A. Flogel. Resolution for Quantified Boolean Formulas. *Information and Computation* 117(1):12–18, 1995.

[Chen, 2004a] H. Chen. Collapsibility and Consistency in Quantified Constraint Satisfaction. *In Proc. of AAAI'04*, pp. 155–160, 2004.

[Chen, 2004b] H. Chen. Quantified Constraint Satisfaction and Bounded Treewidth. *In Proc. of ECAI'04*, pp. 161–165, 2004.

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

[Dechter, 2003] R. Dechter. *Constraint Processing.* Morgan Kaufman, 2003.

[Feder and Kolaitis, 2004] T. Feder and P.G. Kolaitis. Closures and dichotomies for quantified constraints. Manuscript, 2004.

[Gottlob *et al.*, 2000] G. Gottlob, N. Leone, and F. Scarcello. A Comparison of Structural CSP Decomposition Methods. *Artificial Intelligence*, 124(2): 243–282, 2000.

[Gottlob *et al.*, 2001] G. Gottlob, N. Leone, and F. Scarcello. The complexity of acyclic conjunctive queries. *JACM*, 48(3):431–498, 2001.

[Gottlob *et al.*, 2003] G. Gottlob, N. Leone, and F. Scarcello. Robbers, marshals, and guards: game theoretic and logical characterizations of hypertree width. *JCSS*, 66(4):775–808, 2003.

[Grädel, 1999] E. Grädel. On the restraining power of guards. *Journal of Symbolic Logic*, 64:1719–1742, 1999.

[Jeavons *et al.*, 1997] P. Jeavons, D. Cohen, and M. Gyssens. Closure Properties of Constraints. *JACM*, 44(4):527–548, 1997.

[Kolaitis *et al.*, 2000] Ph. G. Kolaitis and M. Y. Vardi. Conjunctive-Query Containment and Constraint Satisfaction. *JCSS*, 61(2):302–332, 2000.

[Mamoulis and Stergiou, 2004] N. Mamoulis and K. Stergiou. Algorithms for Quantified Constraint Satisfaction Problems. *In Proc. of CP'04*, pp. 752–756.

[Robertson and Seymous, 1986] N. Robertson and P.D. Seymour. Graph Minors II. Algorithmic aspects of tree width. *Journal of Algorithms*, 7:309–322, 1986.

[Schaefer, 1978] T.J. Schaefer. The Complexity of Satisfiability Problems *In Proc of. STOC'78*, pp. 216–226, 1978.