CSP Properties for Quantified Constraints: Definitions and Complexity

Lucas Bordeaux
Microsoft Research
J J Thomson Avenue
Cambridge CB3 0FB, UK
lucasb@microsoft.com

Marco Cadoli and **Toni Mancini**
DIS, Università di Roma "La Sapienza"
Via Salaria 113
Rome 00198, ITALY
{cadoli,tmancini}@dis.uniroma1.it

Abstract

Quantified constraints and Quantified Boolean Formulae are typically much more difficult to reason with than classical constraints, because quantifier alternation makes the simple, classical notion of *solution* inappropriate. As a consequence, even such essential CSP properties as consistency or substitutability are not completely understood in the quantified case.

In this paper, we show that most of the properties which are used by solvers for CSP can be generalized to Quantified CSP. We propose a systematic study of the relations which hold between these properties, as well as complexity results regarding the decision of these properties. Finally, and since these problems are typically intractable, we generalise the approach used in CSP and propose weakenings of these notions based on *locality*, which allow for a tractable, albeit incomplete detecting of these properties.

Introduction

Quantified Constraint Satisfaction Problems (QCSP) have recently received increasing attention from the AI community (Bordeaux & Monfroy 2002; Börner *et al.* 2003; Chen 2004a; 2004b; Mamoulis & Stergiou 2004; Gent, Nightingale, & Rowley 2004; Gent, Nightingale, & Stergiou 2005). A large number of solvers are now available for Quantified Boolean Formulae (QBF), which represent the particular case of QCSP where the domains are Boolean and the constraints are clauses, see *e.g.,* (Buening, Karpinski, & Flogel 1995; Cadoli *et al.* 2002; Rintanen 1999) for early papers on the subject, and (Benedetti 2004; Pan & Vardi 2004; Audemard & Saïs 2005) for some of the latest developments. The reason behind this trend is that QCSP and QBF are natural generalisations of CSP and SAT which allow to model a wide range of problems not directly expressible in these formalisms, and with applications in AI and verification.

Quantified constraints are typically much more difficult to reason with than classical constraints. Quantifier alternation makes it much harder to define the usual CSP notions, like consistency of a value for a given variable or substitutability of a value by another, because these properties are based on the notion of *solution* – for instance, a value is (locally) consistent if it participates in a solution to the (sub)problem at hand. In quantified constraints, the value which should be assigned to an existential variable in order to satisfy the constraints depends on the values assigned to the universal variables preceding it, and the flat notion of solution which is used in classical CSPs therefore has to be replaced by a more complex one.

An investigation of the very definition of the CSP notions for quantified constraints is therefore badly needed. This paper addresses this issue and proposes generalisations of consistency, substitutability, and a wider range of CSP properties, to the framework of QCSP. This definitional work is based on a new notion of *outcomes*, which we identify as a key for defining and understanding all these properties. We then classify these properties by studying the relationships between them (*e.g.,* some can be shown to be weaker than others), and we characterise the complexity of their associated decision problem.

Since, as these complexity results show, determining whether the property holds is typically intractable in general, we investigate the use of the same tool which is used in classical CSP, namely *local reasoning*, and we propose local versions of these properties which can be decided in polynomial time.

Note that due to space limitations only some representative or technically challenging proofs are fully developed, the other proofs are either sketched or completely skipped.

Quantified Constraint Satisfaction Problems

In this section, we present all the required material on QCSP. Note that all the results in the paper hold in particular for the particular case of Quantified Boolean Formulae.

Definition of QCSP

Let $\mathbb{D}$ be a finite set. A V-tuple t, where V represents a finite set of variables, is a mapping which associates a value $t_x \in \mathbb{D}$ to every $x \in V$; a V*-relation* is a set of V-tuples.

Definition 1 (QCSP)**.** *A* Quantified Constraint Satisfaction Problem *(QCSP) is a tuple $\phi = \langle X, Q, D, C\rangle$ where: $X = \{x_1, \ldots, x_n\}$ is a linearly ordered, finite set of* variables*; Q associates to each variable $x_i \in X$ a* quantifier *$Q(x_i) \in \{\forall, \exists\}$; D associates to every variable $x_i \in X$ a* domain *$D_{x_i} \subseteq \mathbb{D}$; and C is a finite set of* constraints*, each of which is a V-relation for some $V \subseteq X$.*

The notation $t[x := a]$ will denote the tuple t' defined by $t'_x = a$ and $t'_y = t_y$ for each $y \neq x$. Given a V-tuple t and a subset $U \subseteq V$ of its variables, we denote by $t|_U$ the *restriction* of t to U, which has the same value as t on the variables of U and is undefined elsewhere. An X-tuple t is said to *satisfy* the set of constraints C if $t|_V \in c$ for each V-relation $c \in C$. The set of X-tuples satisfying all constraints of ϕ is denoted by sol^ϕ. We use the following shorthands to denote the set of existential (*resp.* universal) variables, the set of variables of index $\leq j$, and the sets of existential/universal variables of index $\leq j$:

$$\begin{array}{lll} & & X_j = \{x_i \in X \mid i \leq j\} \\ E = \{x_i \in X \mid Q(x_i) = \exists\} & & E_j = E \cap X_j \\ A = \{x_i \in X \mid Q(x_i) = \forall\} & & A_j = A \cap X_j \end{array}$$

A QCSP $\langle X, Q, D, C\rangle$ represents a logical formula F : $Q(x_1)x_1 \in D_{x_1} \dots Q(x_n)x_n \in D_{x_n}$ $(C_1 \wedge \dots \wedge C_m)$. The QCSP is *true* if the interpretation over $\mathbb{D}$ in which all domain and constraint symbols are interpreted according to their definition in D and C is a model of F.

Game-theoretic material

Quantifier alternation is best understood using an adversarial viewpoint, where two players interact. One of them is allowed to choose the values for the existential variables, and its aim is to ultimately make the formula true, while the other assigns the universal variables and aims at falsifying it. Our presentation of this game-theoretic terminology is inspired from (Chen 2004b), which uses a similar notion of winning strategy.

Definition 2 (strategy)**.** *A strategy is a family $\{s_{x_i} \mid x_i \in E\}$ of functions of the following type: for each $x_i \in E$, function s_{x_i} associates to each A_{i-1}-tuple a value in $\mathbb{D}$. This function specifies which value should be assigned to every existential variable depending on the values assigned to the preceding universal variables.*

In particular, if the first k variables of the problem are quantified existentially, we have for every $i \leq k$ a constant $s_{x_i} \in D_{x_i}$ which defines which value should directly be assigned to variable x_i.

Let us insist that the tuple of values that will eventually be assigned to the variables of the problem depends on two things: 1) the strategy, and 2) the sequence of choices of the "adversary", *i.e.*, the values that are assigned to the universal variables. One strategy therefore enables a number of potential *scenarios* to arise, depending on what the adversary will do. These scenarios are defined as follows:

Definition 3 (scenario)**.** *The set of scenarios of a strategy s for a QCSP ϕ, denoted $\mathsf{sce}^\phi(s)$, is the set of X-tuples t which are such that, for each $i \in 1 \dots n$, we have:*

$$\text{if } Q(x_i) = \exists \text{ then } t_{x_i} = s_{x_i}(t|_{A_{i-1}})$$

In other words, the values for the existential variables are determined by the strategy in function of the values assigned to the universal variables preceding it (there is no restriction, on the contrary, on the values assigned to universal variables since we model the viewpoint of the existential player). Of particular interest are the strategies whose scenarios are all solutions. We call them *winning strategies*:

Definition 4 (winning strategy)**.** *A strategy s is a winning strategy for the QCSP ϕ if every scenario $t \in \mathsf{sce}^\phi(s)$ satisfies the constraints of ϕ (in other words: if $\mathsf{sce}^\phi(s) \subseteq \mathsf{sol}^\phi$).*

We denote by WIN^ϕ the set of winning strategies of the QCSP ϕ. It can be shown that a QCSP is true in the sense of the previous subsection (*i.e.*, as a model-checking problem) if it has a winning strategy. Whereas the preceding material is well-known (Chen 2004b), we introduce the new notion:

Definition 5 (outcome)**.** *The set of outcomes of a QCSP ϕ is the set of all scenarios of all its winning strategies, i.e., it is defined as*

$$out^\phi = \{t \mid \exists s \in \mathsf{WIN}^\phi.\ t \in \mathsf{sce}^\phi(s)\}.$$

Note that, whereas $\mathsf{out}^\phi \subseteq \mathsf{sol}^\phi$ in general, the set of outcomes is identical to the set of solutions if all variables are existential. We claim in the following that outcomes are indeed a natural generalisation of the notion of solution, and that they play a similar role in many definitions.

To summarise, we have defined 3 sets of tuples (sol^ϕ: the set of solutions, $\mathsf{sce}^\phi(s)$: the set of scenarios of strategy s, and out^ϕ: the set of outcomes) and one set of strategies, (WIN^ϕ: the set of winning strategies). The superscript ϕ, will from now on be omitted to simplify notation whenever there is no ambiguity. All the notions introduced in this subsection are illustrated in Fig. 1.

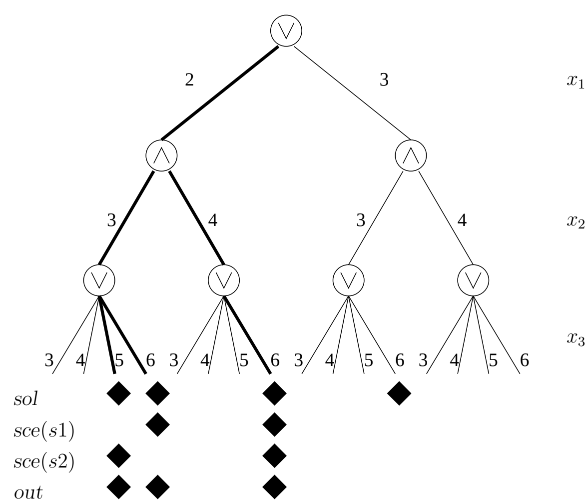

Figure 1: Illustration of the notions of solution, winning strategy, scenario and outcome on the QCSP represented by the logical formula $\exists x_1 \in [2, 3]\ \forall x_2 \in [3, 4]\ \exists x_3 \in [3, 6].\ x_1 + x_2 \leq x_3$. *And* and *or* labels on the nodes correspond to universal and existential quantifiers, respectively. The solutions are all triples $\langle x_1, x_2, x_3\rangle$ *s.t.* $x_1 + x_2 \leq x_3$. The only two winning strategies assign x_1 to 2: one (s_1) assigns x_3 to 6 while the 2nd one (s_2) assigns it to $x_2 + 2$ (note that each strategy is constrained to choose one unique branch for each existential node). The scenarios of s_1 and s_2 are therefore those indicated, while the set of outcomes of the QCSP is the union of the scenarios of s_1 and s_2 (also shown in bold line).

Definitions of the CSP properties

A major part of the CSP literature aims at identifying properties of particular values of some variables. The goal is typically to simplify the problem by ruling out the possibility that a variable x_i can be assigned to a value a. Good reasons for that may be that a is guaranteed not to participate in any solution (a is said *inconsistent* (Mackworth 1977)), that another value b can replace a in any solution involving it (a is *substitutable* to b (Freuder 1991)), or that all solutions involving a can use another value instead (a is *removable*[1]). On the contrary, some other properties can give indications that instantiating x_i to a is a good idea, for instance because no solution can assign it another value (a is said to be *implied* (Monasson *et al.* 1999)), or because we have the guarantee to find a solution with value a on x_i, if a solution exists at all (a is said to be *fixable* for x_i). While all the preceding properties are about values, related variable-oriented notions can be defined: the value assigned to a variable x_i can be forced to a unique possibility (*determined* variable), it can be a function of the values of other variables (*dependent*), or it may not matter at all (*irrelevant*).

In this section, we propose generalisations of the definitions of the main CSP properties to quantified constraints. We will adopt a predicate notation and write, *e.g.*, $p^\phi(x_i, a)$ for the statement "value a has property *p* for variable x_i (in QCSP ϕ)". Here again, the superscript ϕ will be omitted for the sake of clarity, except in the section on *local reasoning* where this non-ambiguous notation will be needed.

Basic definitions

The first definitions we propose (identified by a *d* prefix when an ambiguity with forthcoming definitions is possible) are based on directly rephrasing the original CSP definitions, but using the notion of outcomes in place of solutions:

Definition 6 (properties)**.** *We define:*

$$\begin{aligned}
&\textit{inconsistent}(x_i, a) \equiv \ \forall t \in \mathsf{out}.\ t_{x_i} \neq a\\
&\textit{implied}(x_i, a) \equiv \ \forall t \in \mathsf{out}.\ t_{x_i} = a\\[1ex]
&\textit{d-fixable}(x_i, a) \equiv \ \forall t \in \mathsf{out}.\ t[x_i := a] \in \mathsf{out}\\[1ex]
&\textit{d-substitutable}(x_i, a, b) \equiv\\
&\quad \forall t \in \mathsf{out}.\ t_{x_i} = a \ \rightarrow\ t[x_i := b] \in \mathsf{out}\\[1ex]
&\textit{d-removable}(x_i, a) \equiv\\
&\quad \forall t \in \mathsf{out}.\ t_{x_i} = a \ \rightarrow\ (\exists b \neq a.\ t[x := b] \in \mathsf{out})\\[1ex]
&\textit{d-interchangeable}(x_i, a, b) \equiv\\
&\quad \textit{d-substitutable}(x_i, a, b) \wedge \textit{d-substitutable}(x_i, b, a)\\[1ex]
&\textit{determined}(x_i) \equiv \ \forall t \in \mathsf{out}.\ \forall b \neq t_{x_i}.\ t[x_i := b] \notin \mathsf{out}\\
&\textit{d-irrelevant}(x_i) \equiv \ \forall t \in \mathsf{out}.\ \forall b \in D(x_i).\ t[x_i := b] \in \mathsf{out}\\[1ex]
&\textit{dependent}(V, x_i) \equiv\\
&\quad \forall t, t' \in \mathsf{out}.\ \left(\forall x_j \in V.\ t_{x_j} = t'_{x_j}\right) \ \rightarrow\ t_{x_i} = t'_{x_i}
\end{aligned}$$

[1]The notions of removability and fixability have seemingly been proposed in (Bordeaux, Cadoli, & Mancini 2004). For homogeneity, we adopt the terminology of this paper for all properties.

There would be alternative ways to state the preceding definitions. One can prove in particular that $\forall t \in \mathsf{out}.\ t[x_i := a] \in \underline{\mathsf{out}}$ holds iff $\forall t \in \mathsf{out}.\ t[x_i := a] \in \underline{\mathsf{sol}}$.[2] Fixability could therefore be rewritten under this form, and the other properties could be rephrased in a similar way.

Note also that we obtain the original definitions in the case where all quantifiers are existential (because out = sol). In other words, these are correct generalisations of the classical notions.

Example 1 (illustration of Def. 6)**.** *Consider the QCSP* $\exists x_1 \in [2, 3]\ \forall x_2 \in [3, 4]\ \exists x_3 \in [3, 6].\ x_1 + x_2 \leq x_3$ *(cf. Fig. 1). We have: inconsistent* $(x_1, 3)$*, inconsistent* $(x_3, 3)$*, inconsistent* $(x_3, 4)$*, d-substitutable* $(x_3, 5, 6)$*, d-fixable* $(x_3, 6)$*, d-removable* $(x_3, 5)$*, and implied*$(x_1, 2)$*.*

It is interesting to draw a comparison with what happens if we consider the same problem but with existential quantification ($\exists x_1\ \exists x_2\ \exists x_3.\ x_1 + x_2 \leq x_3$) or, equivalently, if we use the classical properties instead of the quantified ones. Because the tuple $\langle 3, 3, 6\rangle$ is a solution, we have none of the inconsistency, implication, fixability and removability properties. This confirms that the properties we have defined are new notions which do make a difference compared to classical CSP notions. The relation is indeed the following:

Proposition 1. *Let ϕ be a QCSP and let ϕ' be a QCSP obtained by changing a universal quantifier in ϕ to an existential one. Forall x_i, a and b, if property $p(x_i, a)$ (resp. $p(x_i, a, b)$) holds for ϕ', then it also holds for ϕ.*

The classical (unquantified) definitions are therefore correct, *sufficient* conditions for the corresponding quantified property; they allow to detect it only in particular cases.

Generalisation: shallow definitions

The previous definitions are correct in a sense which will be made formal in the next Section, but they are overly restrictive in some cases, as the following example shows:

Example 2 (deep *vs.* shallow definitions)**.** *Consider the QCSP* $\forall x_1 \in [1, 2]\ \exists x_2 \in [3, 4]\ \exists x_3 \in [4, 6].\ x_1 + x_2 = x_3$*. The winning strategies make arbitrary choices for x_2 as long as they give x_3 value $x_1 + x_2$, and the outcomes are the triples $\langle 1, 3, 4\rangle$, $\langle 1, 4, 5\rangle$, $\langle 2, 3, 5\rangle$, $\langle 2, 4, 6\rangle$. Note that for variable x_2, neither values 3 nor 4 are d-fixable, and none is d-substitutable to the other. This somehow goes against the intuition that we are indeed free to choose the value for x_2.*

The reason why our previous definition did not capture this case is that it takes into account the values of the variables occurring *after* the considered variable: values 3 and 4 are interchangeable (for instance) only if the QCSPs resulting from these instantiations can be solved *using the same strategy*. We call these definitions *deep* (hence the *d* prefix). On the contrary, we can formulate *shallow* definitions of the properties, which accept value 4 as a valid substitute for 3

[2]The $\rightarrow$ implication is straightforward. Concerning the other direction ($\leftarrow$), consider the QCSP ϕ' in which all variables $x_1, \ldots, x_{i-1}$ have been instantiated to $t_{x_1}, \ldots, t_{x_{i-1}}$. If $\forall t \in \mathsf{out}.\ t[x_i := a] \in \mathsf{sol}$, then any winning strategy for ϕ' starting with $x_i = t_{x_i}$ can be changed into a winning strategy with $x_i = a$.

because *in any sequence of choices leading to the possibility of choosing 3 for x_2, value 4 is also a valid option*:

Definition 7 (shallow properties)**.** *We define the following "shallow" versions of properties:*

$$\textit{s-fixable}(x_i, a) \equiv$$
$$\forall t \in \mathsf{out}.\ \exists t' \in \mathsf{out}.\ \begin{pmatrix} t|_{X_{i-1}} = t'|_{X_{i-1}} \\ \wedge\ t'_{x_i} = a \end{pmatrix}$$

$$\textit{s-substitutable}(x_i, a, b) \equiv$$
$$\forall t \in \mathsf{out}.\ t_{x_i} = a \ \rightarrow$$
$$\exists t' \in \mathsf{out}.\ \big((t|_{X_{i-1}} = t'|_{X_{i-1}})\ \wedge\ (t'_{x_i} = b)\big)$$

$$\textit{s-removable}(x_i, a) \equiv$$
$$\forall t \in \mathsf{out}.\ t_{x_i} = a \ \rightarrow$$
$$\exists t' \in \mathsf{out}.\ \big(t|_{X_{i-1}} = t'|_{X_{i-1}} \wedge t'_{x_i} \neq a\big)$$

$$\textit{s-interchangeable}(x_i, a, b) \equiv$$
$$\textit{s-substitutable}(x_i, a, b) \wedge \textit{s-substitutable}(x_i, b, a)$$

$$\textit{s-irrelevant}(x_i) \equiv$$
$$\forall t \in \mathsf{out}.\ \forall b \in D(x_i).$$
$$\exists t' \in \mathsf{out}.\ \big((t|_{X_{i-1}} = t'|_{X_{i-1}})\ \wedge\ (t'_{x_i} = b)\big)$$

To see how these definitions compare with classical ones in the case of purely existential CSPs, consider the QCSP $\exists x_1 \in [1, 2]\ \exists x_2 \in [3, 4]\ \exists x_3 \in [4, 6].\ x_1 + x_2 = x_3$. Value 1 is s-substitutable to 2 for x_1, while it is not d-substitutable (*i.e.,* substitutable in the classical sense). The intuition behind this is that here we consider that x_1 is assigned first, and *at this step* the two choices are equivalent. In other words, an additional property holds *because we consider the variables in a particular order* (note that these properties are order-dependent). This relaxed definition of (ordered) substitutability appears to be new and worthy of further study. In particular, it raises the question of determining an instantiation order which reveals as many properties as possible.

Correctness of these definitions and relations between them

Our goal in this section is to prove that the definitions we have introduced are correct. For instance it should be possible to delete inconsistent or removable values without significant alteration of the problem. We start by noticing that, as in the classical case (Bordeaux, Cadoli, & Mancini 2004), the following relations hold between the shallow properties:

Proposition 2. *The following relations hold between the properties (forall x_i, a and b):*

- *inconsistent*$(x_i, a)\ \rightarrow$ $\forall b \in D_{x_i}.$ *d-substitutable*(x_i, a, b)
- *implied*$(x_i, a)\ \leftrightarrow\ \forall b \in D_x \setminus \{a\}.$ *inconsistent*(x_i, b)
- *implied*$(x_i, a)\ \rightarrow$ *d-fixable*(x_i, a)
- $\exists b \in D_x \setminus \{a\}.$ *s-substitutable*$(x_i, a, b)\ \rightarrow$ *s-removable*(x_i, a)
- *inconsistent*$(x_i, a)\ \rightarrow$ *s-removable*(x_i, a)
- *s-fixable*$(x_i, b)\ \leftrightarrow\ \forall a \in D_x.$ *s-substitutable*(x_i, a, b)
- *d-irrelevant*$(x_i)\ \leftrightarrow\ \forall a \in D_x.$ *d-fixable*(x_i, a)
- *s-irrelevant*$(x_i)\ \leftrightarrow\ \forall a \in D_x.$ *s-fixable*(x_i, a)

To complete the picture, we have the following relations between deep and shallow notions (the deep ones are weaker):

Proposition 3. *For each property p among fixability and removability, we have: $\forall x_i\ \forall a.$ d-p$(x_i, a) \rightarrow$ s-p(x_i, a). For each property p among substitutability and interchangeability, we have: $\forall x_i\ \forall a\ \forall b.$ d-p$(x_i, a, b) \rightarrow$ s-p(x_i, a, b).*

Since inconsistency, (deep and shallow) substitutability, and interchangeability are therefore subsumed by the property of removability, we have to prove the correctness of this last notion:

Proposition 4. *Let $\phi = \langle X, Q, D, C\rangle$ be a QCSP in which value $a \in D(x_i)$ is removable for x_i, and let ϕ' denote the same QCSP in which value a is effectively removed (i.e., $\phi' = \langle X, Q, D', C\rangle$ where $D'(x_i) = D(x_i) \setminus \{a\}$ and $D'(x_j) = D(x_j), \forall j \neq i$). Then ϕ is true iff ϕ' is true.*

Proof. (sketch) If ϕ' has a winning strategy then it is straightforward that ϕ also does. On the other hand suppose that ϕ has a winning strategy. That there exist for each $t \in \mathsf{out}$ a $t' \in \mathsf{out}$ *s.t.* $t_{X_{i-1}} = t'_{X_{i-1}}$ and $t_{x_i} \neq a$ means that the QCSP ϕ in which variables $x_1, \ldots, x_i$ are instantiated to $t'_{x_1}, \ldots, t'_{x_i}$ has a winning strategy. Therefore, there exists a winning strategy for ϕ which never assigns value a to x_i, and ϕ' is true. □

Similarly, since implication (shallow or deep) is a special case of fixability, the correctness of the 3 notions is proven by the following proposition:

Proposition 5. *Let $\phi = \langle X, Q, D, C\rangle$ be a QCSP in which value $a \in D(x_i)$ is fixable to x_i, and let ϕ' denote the same QCSP in which value a is effectively fixed (i.e., $\phi' = \langle X, Q, D', C\rangle$ where $D'(x_i) = \{a\}$ and $D'(x_j) = D(x_j), \forall j \neq i$). Then ϕ is true iff ϕ' is true.*

Complexity results

In this section, we study the complexity of the problem of determining whether the properties defined in Definitions 6 and 7 hold. We assume that checking whether $t \in \mathsf{sol}$ (i.e., whether t satisfies the constraints C) can be done in time polynomial in the size of the representation of the input – this assumption holds for the clausal representation used in QBF, for binary CSPs with constraints represented as tables, for numerical constraints, *etc.*

Proposition 6. *Given a QCSP $\phi = \langle X, Q, D, C\rangle$, the problems of deciding whether:*

- *value $a \in D_{x_i}$ is d-fixable, d-removable, inconsistent, implied for variable $x_i \in X$,*
- *value $a \in D_{x_i}$ is d-substitutable to or d-interchangeable with $b \in D_{x_i}$ for variable $x_i \in X$,*
- *variable $x_i \in X$ is dependent on variables $V \subseteq X$, or is d-irrelevant,*

are PSPACE-complete.

Proof. (case of d-fixability) Let us first note that given a tuple t and a QCSP ϕ, determining whether $t \in \mathsf{out}^{\phi}$ can be done in polynomial space using the following algorithm:

```
function isOutcome(t in D1 x ... x Dn) : boolean
begin
  if not(isSolution(t)) then return false;
  forall i = 1..n do
    begin
      if (xi is universally quantified in the QCSP) then
        begin
          forall d in Di such that d <> t[xi] do
            begin
              solve QCSP obtained by fixing variables
                x1..xi-1 to t[x1]..t[xi-1] and xi to d
              if this QCSP is false then return false
            end
        end
    end
  return true
end
```

Iterating through tuples and finding whether one of them (t) exists which is an outcome and which is such that $t[x_i := a]$ is not an outcome can therefore be done in PSPACE.

As for the proof of hardness, we restrict to the boolean case, and reduce the PSPACE-hard problem of deciding whether an arbitrary QBF $\phi \doteq QX\ \gamma(X)$ is true to that of deciding an instance of *deep-fixability*. Let $y \notin X$ be a fresh variable, and let us consider the new QBF $\psi \doteq QX \exists y\ \gamma(X) \wedge \neg y$. We observe that ϕ is true iff ψ is true. We now show that value *true* is *d-fixable* for y in ψ iff ψ is false which, in turn, holds iff ϕ is false.

The proof for the first direction is trivial: if ψ is false, then every value is *d-fixable* for every variable, by definition. As for the other direction instead, if *true* is *d-fixable* for y in ψ, ψ must be false, since, by construction, every winning strategy for ψ would assign *false* to y. □

It is worth noting that such problems are PSPACE-hard even in the case of boolean domains (i.e., QBFs).

An analogous result holds for the shallow properties:

Proposition 7. *Given a QCSP $\phi = \langle X, Q, D, C\rangle$, the problems of deciding whether:*

- *value $a \in D_{x_i}$ is s-fixable, s-removable for variable $x_i \in X$,*
- *value $a \in D_{x_i}$ is s-substitutable to or s-interchangeable with $b \in D_{x_i}$ for variable $x_i \in X$,*
- *variable $x_i \in X$ is s-irrelevant,*

are PSPACE-complete.

Interesting refinements of the results above hold when considering QCSPs with a bounded number of quantifier alternations. We call Σ_kQCSP and Π_kQCSP, respectively, the QCSPs with at most k quantifier alternations starting with an existential or a universal block.

Proposition 8. *Given a Σ_kQCSP $\phi = \langle X, Q, D, C\rangle$, the problems of deciding whether:*

- *value $a \in D_{x_i}$ is deep-/shallow-fixable, deep-/shallow-removable, inconsistent, implied for variable $x_i \in X$,*
- *value $a \in D_{x_i}$ is deep-/shallow-substitutable to or deep-/shallow-interchangeable with $b \in D_{x_i}$ for variable $x_i \in X$,*
- *variable $x_i \in X$ is dependent on variables $V \subseteq X$, or is deep-/shallow-irrelevant,*

are Π_k^p-complete.

Proof. (sketch) Given a Σ_kQCSP ϕ, finding whether a tuple t is in out^{ϕ} is a problem in Π_{k-1}^p (the algorithm used in Prop. 6 gives the main ideas). Checking whether fixability holds (for instance) amounts to check whether a tuple t exists which is in out and such that $t[x_i := a] \notin \mathsf{out}$, a problem which is therefore in $\text{coNP}^{\Pi_{k-1}^p} = \Pi_k^p$. Hardness is shown using a reduction similar to the one in Prop. 6. □

Note that for Π_kQCSPs similar results can be obtained (although slightly less precise ones, the problem being sandwiched between Π_k^p and Π_{k+1}^p).

Local reasoning

The previous section shows that all of the properties we are interested in are computationally difficult to detect – roughly speaking as hard as the resolution of the QCSP problem itself. Following the classical CSP approach, we investigate the use of *local* reasoning to circumvent this intractability by obtaining sufficient (incomplete, or non-necessary) conditions under which the property holds.

Proposition 9. *Let $\phi = \langle X, Q, D, C\rangle$ be a QCSP where $C = \{c_1, \ldots, c_m\}$. We denote by ϕ_k the QCSP $\langle X, Q, D, \{c_k\}\rangle$ in which only the k-th constraint is considered. We have:*

- *for any property p among inconsistency or implication:*

$$\left(\bigvee_{k\in 1..k} \mathsf{p}^{\phi_k}(x_i, a)\right) \ \rightarrow\ \mathsf{p}^{\phi}(x_i, a)$$

- *for any property p among deep or shallow fixability:*

$$\left(\bigwedge_{k\in 1..k} \mathsf{p}^{\phi_k}(x_i, a)\right) \ \rightarrow\ \mathsf{p}^{\phi}(x_i, a)$$

- *for any property p among deep or shallow substitutability or interchangeability:*

$$\left(\bigwedge_{k\in 1..k} \mathsf{p}^{\phi_k}(x_i, a, b)\right) \ \rightarrow\ \mathsf{p}^{\phi}(x_i, a, b)$$

Proof. Common to all these propositions is an important *monotonicity* property of the set of outcomes: if we have two QCSPs $\phi_1 = \langle X, Q, D, C_1\rangle$ and $\phi_2 = \langle X, Q, D, C_2\rangle$ (with the same quantifier prefix) and if the solutions of C_1 are a superset of the solutions of C_2, then $\mathsf{out}^{\phi_1} \supseteq \mathsf{out}^{\phi_2}$. The proof for inconsistency/implication directly follows: for instance (implication) if for some k we have $\forall t \in \mathsf{out}^{\phi_k}.\ t_{x_i} = a$, then it also holds that $\forall t \in \mathsf{out}^{\phi}.\ t_{x_i} = a$.

Consider now deep fixability. Assuming that $\forall k.\ \forall t \in \mathsf{out}^{\phi_k}$ we have $t[x_i := a] \in \mathsf{out}^{\phi_k}$, if we take a tuple $t \in \mathsf{out}^{\phi}$, then $t[x_i := a]$ also belongs to $\mathsf{sol}^{\phi} = \bigcap_k \mathsf{sol}^{\phi_k}$. We already noticed that $\forall t \in \mathsf{out}.\ t[x_i := a] \in \mathsf{sol}$ holds iff $\forall t \in \mathsf{out}.\ t[x_i := a] \in \mathsf{out}$, which completes the proof.

For shallow fixability, assume that $\forall k.\ \forall t \in \mathsf{out}^{\phi_k}.\exists t' \in \mathsf{out}^{\phi_k}.\ t|_{X_{i-1}} = t'|_{X_{i-1}} \wedge t'_{x_i} = a$. Now if some t belongs to

out^{ϕ} it also belongs to each out^{ϕ_k} and for each k there hence exists a tuple $t' \in \mathsf{out}^{\phi_k}$ with $t|_{X_{i-1}} = t'|_{X_{i-1}} \wedge t'_{x_i} = a$. Since all these tuples start the same, this shows the existence of a tuple $t \in \mathsf{out}^{\phi}$ with the same property.

□

Local reasoning can therefore be used to check that some property holds by inspecting the constraints one by one without considering the problem as a whole, for instance that a value is substitutable to another just because this property holds for each constraint. In some other cases like inconsistency, it also allows to determine that a property holds just because one particular constraint has the property.

As already noticed in (Bordeaux, Cadoli, & Mancini 2004) in the non-quantified case, removability is not as well-behaved since it is not possible to use local reasoning.

Conclusions and related work

Defining clearly what basic properties like inconsistency and substitutability mean in the case of quantified constraints is a non-trivial question which is important in understanding and solving this class of problems. A number of definitions were suggested in the literature but many of them did not take into account the specificities of quantifier alternation, which is needed to obtain the most precise definitions. Substitutability in QCSP has for instance been considered by (Mamoulis & Stergiou 2004), but using essentially the classical (unquantified) notions. Most definitions have otherwise been proposed for the particular case of QBF, for instance in (Rintanen 1999; Cadoli *et al.* 2002), several techniques are proposed to fix and remove values. These works have shown that detecting properties is essential and can lead to a consistent pruning of the search space, but no clear and general framework to understand these properties was available.

We have seen in this paper that all basic CSP properties can be generalised to QCSP using the notion of *outcome*. We have first defined the so-called *deep* properties, whose definitions directly follows from the notion of outcome, and we have shown that these notions can be refined into the more general *shallow* ones, giving precise conditions under which a given value can be removed or fixed. Note that the definition of (in)consistency we have obtained is equivalent to the one defined in (Bordeaux & Monfroy 2002); it is nevertheless expressed in a simpler and more elegant way which avoids explicitly dealing with And/Or trees.

Very similarly to the classical CSP case, the decision problem for these properties is no easier than the (Q)CSP problem itself, but we have shown that the same kind of local reasoning used in CSP is also valid for most properties.

References

Audemard, G., and Saïs, L. 2005. A symbolic search-based approach for quantified boolean formulae. In *Proc. of Int. Conf. on Theory and Applications of Satisfiability Testing (SAT)*, to appear. LNCS.

Benedetti, M. 2004. Evaluating QBF via symbolic skolemization. In *Proc. of Int. Conf. on Logic for Programming, Artificial Intelligence and Reasoning (LPAR)*, 285–300. Springer.

Bordeaux, L., and Monfroy, E. 2002. Beyond NP: Arc-consistency for quantified constraints. In *Proc. of Int. Conf. on Principles and Practice of Constraint Programming (CP)*, 371–386. Springer.

Bordeaux, L.; Cadoli, M.; and Mancini, T. 2004. Exploiting fixable, removable and determined values in constraint satisfaction problems. In *Proc. of Int. Conf. on Logic for Programming, Artificial Intelligence and Reasoning (LPAR)*, 270–284. Springer.

Börner, F.; Bulatov, A.; Jeavons, P.; and Krokhin, A. 2003. Quantified constraints: Algorithms and complexity. In *Proc. of Int. Conf. on Computer Science Logic (CSL)*, 58–70. Springer.

Buening, H. K.; Karpinski, M.; and Flogel, A. 1995. Resolution for quantified boolean formulas. *Information and Computation* 117(1):12–18.

Cadoli, M.; Schaerf, M.; Giovanardi, A.; and Giovanardi, M. 2002. An algorithm to evaluate quantified boolean formulae and its experimental evaluation. *J. of Automated Reasoning* 28(2):101–142.

Chen, H. 2004a. Collapsibility and consistency in quantified constraint satisfaction. In *Proc. of Amer. Conf. on Artificial Intelligence (AAAI)*, 155–160. AAAI Press.

Chen, H. 2004b. Quantified constraint satisfaction and bounded treewidth. In *Proc. of Euro. Conf. on Artificial Intelligence (ECAI)*, 161–165. IOS Press.

Freuder, E. C. 1991. Eliminating interchangeable values in constraint satisfaction problems. In *Proc. of Amer. Conf. on Artificial Intelligence (AAAI)*, 227–233. AAAI Press.

Gent, I. P.; Nightingale, P.; and Rowley, A. 2004. Encoding quantified CSPs as quantified boolean formulae. In *Proc. of Euro. Conf. on Artificial Intelligence (ECAI)*, 176–180. IOS Press.

Gent, I.; Nightingale, P.; and Stergiou, K. 2005. QCSP-Solve: A solver for quantified constraint satisfaction problems. In *Proc. of Int. Joint. Conf. on Artificial Intelligence (IJCAI)*, To appear. Morgan Kaufmann.

Mackworth, A. 1977. Consistency in networks of relations. *Artificial Intelligence* 8:99–118.

Mamoulis, N., and Stergiou, K. 2004. Algorithms for quantified constraint satisfaction problems. Technical Report APES-79-2004, Apes research group.

Monasson, R.; Zecchina, R.; Kirkpatrick, S.; Selman, B.; and Troyansky, L. 1999. Determining computational complexity from characteristic 'phase transitions'. *Nature* 400:133–137.

Pan, G., and Vardi, M. Y. 2004. Symbolic decision procedures for QBF. In *Proc. of Int. Conf. on Principles and Practice of Constraint Programming (CP)*, 453–467. Springer.

Rintanen, J. 1999. Improvements to the evaluation of quantified boolean formulae. In *Proc. of Int. Joint. Conf. on Artificial Intelligence (IJCAI)*, 1192–1197. Morgan Kaufmann.