Boolean unification with predicates

SEBASTIAN EBERHARD, STEFAN HETZL and DANIEL WELLER, *Institute of Discrete Mathematics and Geometry, Vienna University of Technology, Wiedner Hauptstrasse 8-10, AT-1040 Wien, Austria.*
Email: eberhard@iam.unibe.ch; stefan.hetzl@tuwien.ac.at; weller@logic.at

Abstract

In this article, we deal with the following problem which we call Boolean unification with predicates: For a given formula $F[X]$ in first-order logic with equality containing an n-ary predicate variable X, is there a quantifier-free formula $G[x_1,\ldots,x_n]$ such that the formula $F[G]$ is valid in first-order logic with equality? We obtain the following results. Boolean unification with predicates for quantifier-free F is Π_2^P-complete. In addition, there exists an EXPTIME algorithm which for an input formula $F[X]$, given as above, constructs a formula G such that $F[G]$ being valid in first-order logic with equality, if such a formula exists. For F of the form $\forall\bar{y}F'[X,\bar{y}]$ with F' quantifier-free, we prove that Boolean unification with predicates is already undecidable. The same holds for F of the form $\exists\bar{y}F'[X,\bar{y}]$ for F' quantifier-free. Instances of Boolean unification with predicates naturally occur in the context of automated theorem proving. Our results are relevant for cut-introduction and the automated search for induction invariants.

Keywords: Boolean unification, second-order unification, quantifier elimination, decidability, complexity.

1 Introduction

Unification is the problem of finding for an input set M of terms $\{t_1,\ldots,t_m\}$ a substitution of the variables $x_1,\ldots,x_n$ occurring in M such that all substituted terms of M are syntactically equal. Such a substitution is called a unifier of $\{t_1,\ldots,t_m\}$. This problem was first formulated and treated by Robinson [19] in the context of resolution. Efficient polynomial time algorithms exist which given an input set $\{t_1,\ldots,t_m\}$ of terms either construct a unifier or prove that no unifier exists (see e.g. Baader and Snyder [4]).

The syntactical unification problem as described above was generalized to equational unification. Here, the problem is to find for an input set M of terms $\{t_1,\ldots,t_m\}$ a substitution of its free variables such that all substituted terms of M are equal with respect to a given equational theory. Equational unification is computationally much more complex than plain unification, for some natural equational theories it is even undecidable (see [4]).

Boolean unification is an important special case of equational unification where the equational theory is given as the theory of Boolean algebras. It was analysed by several authors [2, 5, 16, 17]. Boolean unification has a close connection to propositional logic since finding a unifier for $\{t_1,\ldots,t_m\}$ means to substitute the variables in $\{t_1,\ldots,t_m\}$ such that the substituted terms represent equivalent formulas. The following results for the complexity of Boolean unification have been obtained by Baader in [2]:

(1) Boolean unification is NP-complete if the terms $\{t_1,\ldots,t_m\}$ are built exclusively from variables and the logical symbols $\wedge,\vee,\neg,\top,\bot$.
(2) Boolean unification is Π_2^P-complete if the terms $\{t_1,\ldots,t_m\}$ are allowed to additionally contain free constant symbols.
(3) If the terms $\{t_1,\ldots,t_m\}$ are unrestricted, i.e. if they are allowed to contain free function symbols, then Boolean unification is PSPACE-complete.

Published online September 8, 2015 doi:10.1093/logcom/exv059

We will henceforth refer to the 2nd problem in this list as Boolean unification (BU). In this article, we extend the research on BU by analysing the following more general problem:

PROBLEM (Boolean unification with predicates (BUP))
For an input formula $F[\overline{X}]$ in first-order logic with equality containing predicate variables $\overline{X}$, are there quantifier-free formulas $G_i[x_1,\ldots,x_{k_i}]$ (where k_i is the arity of X_i) such that the formula $F[\overline{G}]$ is valid in first-order logic with equality?

The vector of formulas $\overline{G}$ above will be called a *witness*. Simple instances of BUP are, e.g. the formulas $X(a)\wedge X(b)$ and $X(a)\wedge\neg X(b)$. In the first example, the formula $G[x]:=x\approx a\vee x\approx b$ is a witness, and in the second example, no witness exists since a,b might denote the same object. More complicated instances of BUP naturally occur in the context of proof compression by introducing a cut [13–15]. The instances arising in this context are of the form $\forall x.(X(x)\rightarrow\bigwedge_{t\in T}X(t))\rightarrow F$ (for a set of terms T and quantifier-free F), and a solution to BUP yields a cut-formula that can be used to prove F. Also for the approach in [7], where an algorithm automatically producing induction invariants is presented, the solution of instances of BUP is essential.

The main aim of our article is to analyse the complexity of BUP. As indicated above, BUP contains BU since $\{t_1,\ldots,t_m\}$ is a positive instance of BU iff $t_1\leftrightarrow t_2\leftrightarrow\cdots\leftrightarrow t_m$ is a positive instance of BUP, where the variables of the t_i are considered as 0-ary predicate variables, and the constants of the t_i are considered as 0-ary predicate constants, and therefore BUP is Π_2^P-hard. The first main result of this article will be to show that the following problem, which lies strictly between Boolean unification and BUP, is still in Π_2^P:

PROBLEM (QFBUP)
For a quantifier-free input formula $F[\overline{X}]$ in first-order logic with equality, are there quantifier-free formulas $G_i[x_1,\ldots,x_{k_i}]$ such that the formula $F[\overline{G}]$ is valid in first-order logic with equality?

On the other hand, note that BUP is undecidable since it contains the validity problem for first-order logic with equality. Towards analysing the transition from Π_2^P-completeness to undecidability, we introduce the following restricted versions of BUP.

PROBLEM (EBUP)
For an input formula $F[\overline{X}]=\exists\overline{y}\,H[\overline{X},\overline{y}]$ in first-order logic with equality, where $H[\overline{X},\overline{y}]$ is quantifier-free, are there quantifier-free formulas $G_i[x_1,\ldots,x_{k_i}]$ such that the formula $F[\overline{G}]$ is valid in first-order logic with equality?

PROBLEM (UBUP)
For an input formula $F[\overline{X}]=\forall\overline{y}\,H[\overline{X},\overline{y}]$ in first-order logic with equality, where $H[\overline{X},\overline{y}]$ is quantifier-free, are there quantifier-free formulas $G_i[x_1,\ldots,x_{k_i}]$ such that the formula $F[\overline{G}]$ is valid in first-order logic with equality?

THEOREM 1
EBUP is undecidable.

PROOF. We give a computable reduction from the validity problem of first-order logic with equality to EBUP. First, note that this validity problem reduces to the validity problem for formulas of the form $\exists\overline{x}F$, with F quantifier-free: let F be an arbitrary first-order formula, then we can compute a formula $\exists x_1\forall y_1\ldots\exists x_n\forall y_nF'(\overline{x},\overline{y})$ with $F'(\overline{x},\overline{y})$ quantifier-free that is logically equivalent to F by prenexification. Let $f_1,\ldots,f_n$ be functions symbols not occuring in F' and define $t_i:=f_i(x_1,\ldots,x_i)$. Then $\exists x_1\cdots x_nF'(\overline{x},t_1,\ldots,t_n)$ is valid exactly if F is valid (since Skolemization of universal quantifiers

preserves validity). Finally, if EBUP is decidable, then we can decide the validity problem for $\exists \overline{x} F$ by giving it as input to the Turing machine deciding EBUP. ■

On the other hand, the validity problem for formulas of the form $\forall \overline{x} G$, with G quantifier-free, is decidable, so this proof does not apply to UBUP. The second main aim of this article is therefore the development of an undecidability proof for UBUP by reduction from a different well-known problem, the *Post Correspondence Problem*.

2 Related research

2.1 Second-order unification

Instead of working modulo an equational theory, the standard (i.e. syntactic first-order) unification problem has also been extended to allow for higher-order variables. *Second-order unification* permits, in addition to individual variables, also function variables and asks for a substitution resulting in syntactic equality of the terms to be unified. An n-ary function variable can then be substituted by a term of the form $\lambda x_1 \ldots x_n.t$ where the β-reduction is considered part of the substitution. The term t is built from the original signature and may contain any number of occurrences of any of the x_i. This problem has been shown to be undecidable in [12].

Context unification is the restriction of this problem to terms $\lambda x.t$ where t is still in the original signature but — in contrast to second-order unification — may contain only one occurrence of x. This restriction of permitting at most one occurrence of x turns the second-order variables into context variables (into which a substitution inserts a context, i.e. a term with a hole). Several subclasses of context unification are known to be decidable, see e.g. [20, 21]. In particular, the restriction of this problem to the class containing at most one context variable has been shown NP-complete in [11].

The second-order aspect of these problems is conceptually similar to the problems treated in this article: the problem QFBUP could be described as a second-order unification problem modulo logical equivalence. But in the light of results on first-order equational unification, see e.g. [4], it is not surprising that working modulo logical equivalence instead of syntactical equality constitutes an essential difference, and hence the techniques and results from second-order and context unification are not applicable to the problems studied in this article.

2.2 Quantifier elimination

If L is a language and T a theory in L, we say that T has *quantifier elimination* if for every formula F in L, there exists a quantifier-free formula G in L such that $F \leftrightarrow G$ holds in T. Clearly, quantifier elimination is conceptually related to the BUP problem: if, e.g. we are able to solve the BUP problem for $F[\overline{X}]$ by providing quantifier-free formulas $\overline{G}$ such that $\exists \overline{X}\, F[\overline{X}] \leftrightarrow F[\overline{G}]$ is valid, then this also yields a method for the elimination of second-order predicate quantifiers.

On the other hand, quantifier elimination does not necessarily yield witnesses: the first-order theory of real closed fields in the language $L = \{=, \leq, +, -, \cdot, 0, 1\}$, e.g. famously has first-order quantifier elimination, but the valid formula $\exists x\, x \cdot x = 1 + 1$ does not have a witness since there is no term of L representing $\sqrt{2}$.

Still, some quantifier elimination methods may yield the witnesses we require in our context. Since we are looking for witnesses of predicate variables, the most relevant quantifier elimination procedures are those for second-order predicate quantifiers. The SCAN algorithm introduced in [9] (see also [10]) is such a quantifier elimination algorithm based on the resolution calculus. SCAN

is known to be correct but incomplete on the class of formulas $\exists X_1 \cdots X_n F$, where F is a first-order formula. SCAN improves a similar algorithm introduced by Ackermann in [1].

More recently, the DLS algorithm was developed [6] (see also [10]). Analogously to the SCAN algorithm, the DLS algorithm computes, given a second-order formula, a logically equivalent formula that is free of second-order quantifiers if it terminates succesfully. In contrast to the SCAN algorithm, DLS does not involve the construction of a resolution derivation, but is based on a more direct application of a central lemma of [1]. It turns out that in the context of QFBUP, the DLS algorithm always terminates successfully, and that it can be used to compute the witnesses we require. Hence, we will use (a specialization of) the DLS algorithm for obtaining the Π_2^P-completeness result for QFBUP in Section 5.1.

3 Outline of the article

We start by recalling basic definitions of logic relevant to our problems in Section 4. We then turn our attention to the QFBUP problem: in Section 5.1, we give an EXPTIME algorithm which from an input instance $\mathcal{P}$ of QFBUP produces a witness if it exists, thereby reducing the QFBUP problem to the validity problem of formulas of the form $\exists \overline{X}\, F[\overline{X}]$ with $F[\overline{X}]$ quantifier-free. We complete the study of QFBUP by determining the complexity of this validity problem in Section 5.2. Finally, we turn to UBUP and show its undecidability in Section 6 by a reduction from Post's Correspondence Problem.

4 Preliminaries

The logical formalism we work in is that of classical first-order logic with equality extended by quantification over predicates. More precisely, we assume given a language $\mathcal{L}$ containing, for every $n \in \mathbb{N}$, countably many function and predicate symbols of arity n. In particular, we assume that $\mathcal{L}$ contains a distinguished binary predicate symbol for equality denoted by $\approx$. *Terms* are defined as usual from individual variables, usually denoted by $x, y, z, \ldots$, and function symbols from $\mathcal{L}$. *Atomic formulas* are defined as usual from terms, the predicate symbols from $\mathcal{L}$, as well as n-ary predicate variables, usually denoted by $X, Y, Z, \ldots$, for every $n \in \mathbb{N}$. *Formulas* are defined as usual from atomic formulas, the propositional connectives $\wedge, \vee, \neg$, as well as first- and second-order quantifiers $\exists x, \exists X$. A formula or term is called *ground* if it does not contain variables. The *size* of a formula is defined as the number of symbols it contains.

We will use the following notation to indicate substitution: if we introduce a formula as $F[x_1, \ldots, x_n]$ and first-order terms $t_1, \ldots, t_n$, then by $F[t_1, \ldots, t_n]$ we denote the formula $F[x_1, \ldots, x_n]$ after simultaneous substitution of x_i by t_i. For a k-ary predicate variable X, if we introduce a formula as $F[X]$ and another formula as $G[x_1, \ldots, x_k]$, then by $F[G]$ we denote the formula obtained from $F[X]$ by substituting $X(t_1, \ldots, t_k)$ by $G[t_1, \ldots, t_k]$. An analogous notation is used for formulas introduced as $F[X_1, \ldots, X_n]$.

A *structure* is a pair $\mathcal{M} = (M, I)$ where M is a set and I is an interpretation of $\mathcal{L}$, i.e. $I(P) \subseteq M^k$ for k-ary predicates $P \in \mathcal{L}$, and $I(f) : M^k \to M$ for k-ary function symbols $f \in \mathcal{L}$. In particular, for 0-ary predicates P, $I(P)$ is either the empty set, which we denote by $I(P) = \bot$, or $I(P)$ is the singleton set containing the empty tuple, which we denote by $I(P) = \top$. An *environment* is an interpretation of the set of variables. If θ is an interpretation, x a variable, and $m \in M$, then $\theta[x := m]$ is defined by $\theta[x := m](x) = m$ and $\theta[x := m](y) = \theta(y)$ for $x \neq y$. $\theta[X := S]$ is defined analogously for k-ary predicate variables X and $S \subseteq M^k$. For a structure $\mathcal{M} = (M, I)$, an environment θ, and a formula F,

the relation $\mathcal{M},\theta \models F$ is defined as usual. In particular, for a k-ary predicate variable X, we have the clause $\mathcal{M},\theta \models \exists X\, F[X]$ iff there is a $S \subseteq M^k$ such that $\mathcal{M},\theta[X := S] \models F[X]$. We define $\mathcal{M} \models F$ iff $\mathcal{M},\theta \models F$ for all environments θ, and say that F *holds in* $\mathcal{M}$.

A formula F is *valid in first-order logic without equality* if $\mathcal{M} \models F$ for all structures $\mathcal{M}$. We say that $\mathcal{M} = (M, I)$ is a *structure for first-order logic with equality* if $I(\approx) = \{(m,m) \mid m \in M\}$. A formula F then is *valid in first-order logic with equality* if $\mathcal{M} \models F$ for all structures for first-order logic with equality $\mathcal{M}$. If it is clear from the context which notion of validity we refer to, we will simply write 'structure' and 'F is valid'.

5 The quantifier-free Boolean unification problem

The complexity-theoretic characterization of QFBUP will be obtained in two steps:

THEOREM 2
There exists an EXPTIME function wit from quantifier-free formulas to quantifier-free formulas such that

$$(\exists X\, F[X]) \leftrightarrow F[\mathsf{wit}(F[X])]$$

is valid in first-order logic with equality for all quantifier-free formulas $F[X]$.

THEOREM 3
QFBUP is Π_2^P-complete.

The proofs of these results are presented in the two subsequent sections.

5.1 The DLS′ *algorithm*

In this section, 'valid' always means 'valid in first-order logic with equality.' The DLS algorithm, see [10], is a quantifier elimination algorithm for formulas $\exists X\, F[X]$, where F may be an arbitrary first-order formula. If it terminates successfully on such a formula, it yields a first-order formula G such that $\exists X\, F[X] \leftrightarrow G$ is valid. Here, we study the specialization DLS′ of the DLS algorithm to the setting of formulas $\exists X\, F[X]$ with $F[X]$ quantifier-free, which is defined as follows:

(1) Given such a formula $\exists X\, F[X]$, compute a DNF $C_1[X] \vee \cdots \vee C_n[X]$ of $F[X]$.
(2) Write each $C_i[X]$ in the form $\alpha_i[X] \wedge \beta_i[X]$, where $\alpha_i[X]$ contains exactly the positive occurrences of X in $C_i[X]$.
(3) Let $\alpha_i[X]$ be $X(\bar{t}_1) \wedge \cdots \wedge X(\bar{t}_k)$. Define $G_i[\bar{x}] := \bar{x} \approx \bar{t}_1 \vee \cdots \vee \bar{x} \approx \bar{t}_k$ (where $\bar{t} \approx \bar{s} := t_1 \approx s_1 \wedge \cdots \wedge t_m \approx s_m$ as usual).
(4) Output

$$\begin{aligned} S[\bar{x}] := {} & (\beta_1[G_1] \rightarrow G_1[\bar{x}]) \wedge \\ & (\neg\beta_1[G_1] \wedge \beta_2[G_2] \rightarrow G_2[\bar{x}]) \wedge \\ & \cdots \\ & (\neg\beta_1[G_1] \wedge \cdots \wedge \neg\beta_{n-1}[G_{n-1}] \wedge \beta_n[G_n] \rightarrow G_n[\bar{x}]). \end{aligned}$$

Let us consider an examplary run of the algorithm.

EXAMPLE 4
Consider the formula $\exists X\, F[X]$ with

$$F[X] = ([X(a) \wedge \neg X(f(b))] \vee [X(f(b)) \wedge \neg X(a)]).$$

$F[X]$ is already in a DNF of the form $C_1[X] \vee C_2[X]$. We have $G_1[x] = x \approx a$ and $G_2[x] = x \approx f(b)$, and therefore

$$S[x] = (\neg f(b) \approx a \rightarrow x \approx a) \wedge (\neg\neg f(b) \approx a \wedge \neg a \approx f(b) \rightarrow x \approx f(b))$$

It is easily seen that in first-order logic with equality, the equivalences

$$F[S] \leftrightarrow \neg f(b) \approx a \leftrightarrow \exists X\, F[X]$$

are valid.

To prove Theorem 2, it suffices to prove the following.

THEOREM 5
The DLS′ algorithm terminates on every input $\exists X\, F[X]$ in exponential time in the size of $F[X]$ and outputs a formula $S[\overline{x}]$ such that $\exists X\, F[X] \leftrightarrow F[S]$ is valid.

Towards the proof of this result, we need the following two results. The first is a specialization of Ackermann's Lemma [1, 10]:

LEMMA 6
Let $A[\overline{x}], B[X]$ be formulas such that $A[\overline{x}]$ contains no occurrences of X, and $B[X]$ contains only negative occurrences of X. Then the following is valid:

$$\exists X\, (\forall \overline{x}(A[\overline{x}] \rightarrow X(\overline{x})) \wedge B[X]) \leftrightarrow B[A].$$

The second result shows how to obtain witnesses for the individual disjuncts of the DNF.

LEMMA 7
Let $C_i[X], \beta_i[X], G_i[\overline{x}]$ be as above. Then the following are valid:

$$C_i[G_i] \leftrightarrow \beta_i[G_i] \text{ and } \exists X\, C_i[X] \leftrightarrow C_i[G_i].$$

PROOF. Note that $C_i[G_i] \leftrightarrow \beta_i[G_i]$ is valid since $\alpha_i[G_i]$ is valid. Furthermore, we have validity of

$$\alpha_i[X] \leftrightarrow \forall \overline{x}(G_i[\overline{x}] \rightarrow X(\overline{x}))$$

and hence by Lemma 6

$$\exists X\, C_i[X] \leftrightarrow \exists X\, (\forall \overline{x}(G_i[\overline{x}] \rightarrow X(\overline{x})) \wedge \beta_i[X]) \leftrightarrow \beta_i[G_i].$$

■

It remains to show that the combination $S[\overline{x}]$ of these single witnesses are a witness for the whole disjunction.

PROOF OF THEOREM 5
Assume $\mathcal{M} \models \exists X\, F[X]$. Then

$$\mathcal{M} \models (\exists X\, C_1[X]) \vee \cdots \vee (\exists X\, C_n[X])$$

by logic. Let $i \in \{1,\ldots,n\}$ be the least such that $\mathcal{M} \models \exists X\, C_i[X]$. Then for all $j < i$ we have $\mathcal{M} \models \neg\exists X\, C_j[X]$ and hence by Lemma 7

$$(*) \quad \mathcal{M} \models \neg\beta_1[G_1] \wedge \cdots \wedge \neg\beta_{i-1}[G_{i-1}] \wedge \beta_i[G_i].$$

We claim that $\mathcal{M} \models \forall\overline{x}(S[\overline{x}] \leftrightarrow G_i[\overline{x}])$:

(1) If $\mathcal{M} \models S[\overline{x}]$, then $\mathcal{M} \models G_i[\overline{x}]$ by $(*)$.
(2) Assume $\mathcal{M} \models G_i[\overline{x}]$. Note that all antecedents in the conjuncts of $S[\overline{x}]$ are false in $\mathcal{M}$ except for the i'th antecendent. The i'th succedent is also true in $\mathcal{M}$ by assumption, hence $\mathcal{M} \models S[\overline{x}]$.

Hence by Lemma 7 we have $\mathcal{M} \models \exists X\, C_i[X] \Rightarrow \mathcal{M} \models C_i[G_i] \Rightarrow \mathcal{M} \models C_i[S] \Rightarrow \mathcal{M} \models F[S]$. The exponential run-time of DLS' is trivial (all steps take at most quadratic time in the size of a DNF of $F[X]$). ■

We formulate the following corollary, since the fact that a *single witness* suffices to establish validity of an existential formula is a well-known property in logic (see e.g. [8]).

COROLLARY 8
The set $\mathcal{F}$ of formulas of the form $\exists X\, F[X]$ for a quantifier-free formula $F[X]$ has the EXPTIME existence property; i.e. if $\exists X\, F[X]$ is valid, then a witness of F can be computed in exponential time in $F[X]$.

5.2 Π_2^P-*completeness of* QFBUP

The aim of this section is to establish Theorem 3, i.e. the complexity theoretic characterization of the QFBUP problem. QFBUP is Π_2^P-hard: the hardness proof of BUP from Section 1, showing that BU is contained in BUP, applies already to QFBUP.

It therefore suffices for Π_2^P-completeness to give a polynomial time reduction to a problem which is in Π_2^P. We choose the validity problem of *quantified boolean formulas (QBFs)*: a formula F is called a *QBF* if F contains no predicate constants, no first-order quantifiers, and all predicate variables in F are 0-ary. Then the following problem is well-known to be Π_2^P-complete (see [18]):

PROBLEM (Π_2−TQBF)
Given a closed QBF $F = \forall\overline{X}\exists\overline{Y}\, G[\overline{X},\overline{Y}]$, with $G[\overline{X},\overline{Y}]$ quantifier-free, is F valid?

Note that for QBFs, the notions of validity with/without equality coincide since formulas do not contain $\approx$. Furthermore, if F is a QBF and $(M,I),\theta \models F$ then $(M',I'),\theta \models F$ for any set M' and any interpretation I' since there are no terms, no predicate constants nor first-order variables in F. Hence we will always simply write $\theta \models F$ for QBFs F.

Towards our reduction, we introduce two validity problems corresponding to QFBUP: by $2\mathrm{QFV}_{\approx}$ we will denote the problem of determining, given a quantifier-free formula $F[\overline{X}]$, whether $\exists\overline{X}\, F[\overline{X}]$ is valid in first-order logic with equality. By 2QFV we denote the analogous problem for first-order logic without equality.

COROLLARY 9
QFBUP is p-equivalent to 2QFV$_\approx$.

PROOF. Let $F[\overline{X}]$ be a positive instance of QFBUP, i.e., there are $G_1,\ldots,G_n$ s.t. $F[\overline{G}]$ is valid in first-order logic with equality. Then also the formula $\exists\overline{X}\,F[\overline{X}]$ is valid and hence $F[\overline{X}]$ is a positive instance of 2QFV$_\approx$.

For the other direction, let $F[\overline{X}]$ be a positive instance of 2QFV$_\approx$, then $\exists\overline{X}\,F[\overline{X}]$ is valid and by repreated application of Theorem 5 we obtain formulas $G_1,\ldots,G_n$ s.t. $\exists\overline{X}\,F[\overline{X}]\leftrightarrow F[\overline{G}]$ is valid. Hence $F[\overline{G}]$ is valid in first-order logic with equality, i.e., $F[\overline{X}]$ is a positive instance of QFBUP. ■

Let us introduce some notions that will be used in the following proofs. Let R,S be sets of atomic formulas, and let $\tau:S\to R$. By abuse of notation, we denote by τ also the map from formulas over S to formulas over R defined by $\tau(A\circ B)=\tau(A)\circ\tau(B)$ for propositional connectives $\circ$. We will often make use of the fact that if $\mathcal{M},\mathcal{N}$ are structures, then $\mathcal{M},\sigma\models A$ iff $\mathcal{N},\theta\models\tau(A)$ for all $A\in S$ implies $\mathcal{M},\sigma\models B$ iff $\mathcal{N},\theta\models\tau(B)$ for all formulas B over S.

For predicate symbols P, function symbols f, vectors of terms $\bar{s},\bar{t}$, and terms r,u,v, define the formulas

$$\begin{aligned}
\mathrm{EqAx}_1(P,\bar{t},\bar{s}) &:= \bar{t}\approx\bar{s}\wedge P(\bar{t})\to P(\bar{s})\\
\mathrm{EqAx}_2(f,\bar{t},\bar{s}) &:= \bar{t}\approx\bar{s}\to f(\bar{t})\approx f(\bar{s})\\
\mathrm{EqAx}_3(r,u,v) &:= r\approx u\wedge u\approx v\to r\approx v\\
\mathrm{EqAx}_4(r,u) &:= r\approx u\to u\approx r\\
\mathrm{EqAx}_5(r) &:= r\approx r.
\end{aligned}$$

It is an elementary result of logic that a formula F is valid in first-order logic with equality iff $\mathrm{EqAx}\to F$ is valid in first-order logic without equality, where EqAx is a conjunction of universal closures of instances of the formulas given above. The following result is proven by strengthening this observation for the class of formulas under consideration: in our setting, it is not necessary to use the universal closure, and the number of instances required is polynomially bounded.

LEMMA 10
2QFV$_\approx$ ptime reduces to 2QFV.

PROOF. Fix a quantifier-free formula $F[X]$. Denote by Rel the set of predicate symbols occuring in $F[X]$ (including the predicate variable X as well as the symbol $\approx$ in case it does not occur in $F[X]$) and by Fun the function symbols occuring in $F[X]$. Furthermore set $\mathrm{Terms}:=\{t\mid t \text{ a term in } F[X]\}$, and denote by At the set of atomic formulas built from Terms and Rel. Define

$$\mathrm{EqAx}:=\bigwedge_{\substack{P\in\mathrm{Rel},\\ f\in\mathrm{Fun},\\ \bar{t},\bar{s}\in\mathrm{Terms}^n,\\ r,u,v\in\mathrm{Terms}}}\mathrm{EqAx}_1(P,\bar{t},\bar{s})\wedge\mathrm{EqAx}_2(f,\bar{t},\bar{s})\wedge\mathrm{EqAx}_3(r,u,v)\wedge\mathrm{EqAx}_4(r,u)\wedge\mathrm{EqAx}_5(r).$$

Note that the formula $F'[X]:\mathrm{EqAx}\to F[X]$ can be computed in polynomial time in $F[X]$. We will show that $\exists X\,F[X]$ is valid in first-order logic with equality iff $\exists X\,F'[X]$ is valid in first-order logic without equality.

Let $\mathcal{M} \not\models \exists X\, F[X]$. It suffices to show that $\mathcal{M} \models (\forall X\, \mathrm{EqAx}) \wedge (\forall X\, \neg F[X])$. Since $\mathcal{M}$ interprets $\approx$ as real equality on M, we have $\mathcal{M} \models \forall X\, \mathrm{EqAx}$, and since $\mathcal{M}$ is a countermodel to $\exists X\, F[X]$, we have $\mathcal{M} \models \forall X\, \neg F[X]$.

Now let $\mathcal{M} = (M, I)$ be such that $\mathcal{M}, \sigma \not\models \exists X(\mathrm{EqAx} \to F[X])$. Define an equivalence relation $\sim$ on Terms by $s \sim t :\Leftrightarrow \mathcal{M} \models s \approx t$ and a structure $\mathcal{N} = (N, J)$ where $N := \{[t]_\sim \mid t \in \mathrm{Terms}\}$ and for f a function symbol of arity n and P a predicate symbol of arity n (where c is some arbitrary but fixed constant symbol from $F[X]$ which we w.l.o.g. assume to exist)

$$J(f) := [t_1]_\sim, \ldots, [t_n]_\sim \mapsto \begin{cases} [f(t_1, \ldots, t_n)]_\sim & \text{if } f(t_1, \ldots, t_n) \in \mathrm{Terms}, \\ [c]_\sim & \text{if } f(t_1, \ldots, t_n) \notin \mathrm{Terms} \end{cases}$$
$$J(P) := \{([t_1]_\sim, \ldots, [t_n]_\sim) \mid \mathcal{M}, \sigma \models P(t_1, \ldots, t_n)\}.$$

Note that, since $\mathcal{M}, \sigma \models \forall X\, \mathrm{EqAx}$, $J(f)$ is well-defined for all $f \in \mathrm{Fun}$. Furthermore, $J(\approx)$ is real equality on N. It remains to show that $\mathcal{N}, \theta \models \forall X\, \neg F[X]$, where θ is an arbitrary environment. For a contradiction, assume that $\mathcal{N}, \theta[X := S] \models F[X]$ for some $S = \{([t_{i,1}]_\sim, \ldots, [t_{i,n}]_\sim) \mid 1 \leq i \leq k\}$. Define $R := \{(I(t_{i,1}), \ldots, I(t_{i,n})) \mid 1 \leq i \leq k\}$, then $\mathcal{M}, \sigma[X := R] \models F[X]$ iff $\mathcal{N}, \theta[X := S] \models F[X]$:

(1) $\mathcal{M}, \sigma[X := R] \models P(t_1, \ldots, t_n) \Leftrightarrow ([t_1]_\sim, \ldots, [t_n]_\sim) \in J(P) \Leftrightarrow \mathcal{N}, \theta[X := S] \models P(t_1, \ldots, t_n)$,
(2) $\mathcal{M}, \sigma[X := R] \models X(t_1, \ldots, t_n) \Leftrightarrow (I(t_1), \ldots, I(t_n)) \in R \Leftrightarrow ([t_1]_\sim, \ldots, [t_n]_\sim) \in S \Leftrightarrow \mathcal{N}, \theta[X := S] \models X(t_1, \ldots, t_n)$.

Hence $\mathcal{M}, \sigma \models \exists X\, F[X]$, contradiction. ■

EXAMPLE 11
Consider the formula $a \approx b$. This formula is not valid in first-order logic with equality; the structure (M, I) with $M = \{a, b\}$ and $I(\approx) = \{(a, a), (b, b)\}$ is a countermodel. The formula EqAx constructed in the proof of Lemma 10 is equivalent to (in first-order logic without equality) $(a \approx b \leftrightarrow b \approx a) \wedge a \approx a \wedge b \approx b$. Note that the structure given above is also a countermodel for $\mathrm{EqAx} \to a \approx b$.

LEMMA 12
2QFV ptime reduces to Π_2−TQBF.

PROOF. Fix a quantifier-free formula $G[X]$, denote by $\mathrm{At} := \mathrm{ConsAt} \cup \mathrm{VarAt}$ the set of atomic formulas in $G[X]$, where ConsAt contains the atoms with constant head and VarAt contains the atoms with variable head, and fix any injection τ from At to the set of 0-ary predicate variables. Let $\tau(\mathrm{ConsAt}) = \{Y_1, \ldots, Y_n\}$ and $\tau(\mathrm{VarAt}) = \{Z_1, \ldots, Z_m\}$. We show that $\exists X\, G[X]$ is valid iff $\forall Y_1 \cdots \forall Y_n \exists Z_1 \cdots \exists Z_m\, \tau(G[X])$ is valid.

Let $\mathcal{M} = (M, I)$ be a countermodel of $\exists X\, G[X]$. It suffices to give an interpretation J of the Y_i such that $J \models \forall Z_1 \cdots \forall Z_m \neg \tau(G[X])$. Define

$$J(Y_i) = \begin{cases} \top \text{ if } \mathcal{M} \models \tau^{-1}(Y_i), \\ \bot \text{ otherwise.} \end{cases}$$

For contradiction, assume that $J \models \exists Z_1 \cdots \exists Z_m\, \tau(G[X])$, i.e. there exists an extension J' of J sucht that $J' \models \tau(G[X])$. Let $\tau^{-1}(Z_i) = X(t_{1,i}, \ldots, t_{k,i})$ and define $S := \{(I(t_{1,i}), \ldots, I(t_{k,i})) \mid J'(Z_i) = \top\}$. Then $\mathcal{M}[X := S] \models G[X]$ iff $J' \models \tau(G[X])$:

(1) Let $\tau(P(t_1, \ldots, t_n)) = Y_k$. Then $\mathcal{M}[X := S] \models P(t_1, \ldots, t_n) \Leftrightarrow J(Y_k) = \top \Leftrightarrow J' \models Y_k$.
(2) Let $\tau(X(t_1, \ldots, t_n)) = Z_k$. Then $\mathcal{M}[X := S] \models X(t_1, \ldots, t_n) \Leftrightarrow (I(t_1), \ldots, I(t_n)) \in S \Leftrightarrow J'(Z_k) = \top \Leftrightarrow J' \models Z_k$.

Hence $\mathcal{M}[X := S] \models G[X]$, contradiction.

For the other direction, assume that there exists a function $I : \{Y_1, \ldots, Y_n\} \to \{\top, \bot\}$ such that $I \models \forall Z_1 \cdots \forall Z_m \neg\tau(G[X])$. Let Terms be the set of terms occuring in $G[X]$, and define the interpretation

$$J(s) := \begin{cases} (t_1, \ldots, t_n) \mapsto s(t_1, \ldots, t_n) & \text{if } s \text{ a function symbol of arity } n, \\ \{(t_1, \ldots, t_n) \mid I \models \tau(s(t_1, \ldots, t_n))\} & \text{if } s \text{ a predicate symbol of arity } n. \end{cases}$$

Then $\mathcal{M} := (\text{Terms}, J)$ is a structure such that $J(t) = t$ for all $t \in \text{Terms}$. We claim that $\mathcal{M} \models \forall X\, \neg G[X]$. For contradiction, assume that $\mathcal{M}[X := S] \models G[X]$ for some $S \subseteq \text{Terms}^k$. Define

$$I' : \begin{cases} Y_i & \mapsto I(Y_i) \\ \tau(X(t_1, \ldots, t_k)) & \mapsto \top \text{ if } (t_1, \ldots, t_k) \in S \\ \tau(X(t_1, \ldots, t_k)) & \mapsto \bot \text{ if } (t_1, \ldots, t_k) \notin S. \end{cases}$$

Then $\mathcal{M}[X := S] \models G[X]$ iff $I' \models \tau(G[X])$:

(1) Let $\tau(P(t_1, \ldots, t_n)) = Y_k$. Then $\mathcal{M}[X := S] \models P(t_1, \ldots, t_n) \Leftrightarrow (J(t_1), \ldots, J(t_n)) \in J(P) \Leftrightarrow (t_1, \ldots, t_n) \in \{(t_1, \ldots, t_n) \mid I \models \tau(P(t_1, \ldots, t_n))\} \Leftrightarrow I \models Y_k$.

(2) Let $\tau(X(t_1, \ldots, t_n)) = Z_k$. Then $\mathcal{M}[X := S] \models X(t_1, \ldots, t_n) \Leftrightarrow (J(t_1), \ldots, J(t_n)) \in S \Leftrightarrow I' \models Z_k$.

Therefore $I' \models \tau(G[X])$, contradiction. ■

EXAMPLE 13
Consider the formula $X(a) \leftrightarrow P(a)$. Let Y, Z be a 0-ary predicate variables and let $\tau(X(a)) = Y$ and $\tau(P(a)) = Z$. Then $\exists X(X(a) \leftrightarrow P(a))$ and $\forall Y \exists Z(Z \leftrightarrow Y)$ are both valid; in the first case we may take the witness $G[x] := P(x)$, and in the second case we may take Y as the witness for Z.

Lemmas 10 and 12 complete the proof of Theorem 3. In fact, we have shown that all problems considered in this section are Π_2^P-complete:

THEOREM 14
QFBUP, $2\text{QFV}_{\approx}$, and 2QFV are all Π_2^P-complete.

6 The universal Boolean unification problem

Having established the decidability and complexity of the quantifier-free BUP problem in the previous section, as well as the undecidability of the BUP problem when restricted to purely existential formulas in Section 1, we now turn to the investigation of the BUP problem for purely universal formulas, i.e. the UBUP problem. Our aim is to prove that this problem is undecidable as well. Our proof will proceed by reducing the modified Post's Correspondence Problem[1] (in the following: PCP) to UBUP by computing, from an instance $\mathcal{P}$ of PCP, a quantifier-free formula $F_{\mathcal{P}}[X, \bar{y}]$ such that $\forall \bar{y} F_{\mathcal{P}}[X, \bar{y}]$ has a witness exactly if $\mathcal{P}$ has a solution. Note that a single predicate variable X in $F_{\mathcal{P}}[X, \bar{y}]$ is sufficient for undecidability.

Let us begin by defining PCP in detail.

[1] Modified Post's Correspondence Problem differs from the 'usual' Post's Correspondence Problem by requiring solutions to end with index 1, see [22, section 5.2].

DEFINITION 15
A *binary word* is a (possibly empty) string over the alphabet $\{0,1\}$. The empty string is denoted by ϵ. For strings $s_1 = a_1 \cdots a_n, s_2 = b_1 \cdots b_m$, we define the *concatenation* $s_1 \circ s_2 := a_1 \cdots a_n b_1 \cdots b_m$. In the following, we will simply write $s_1 s_2$ instead of $s_1 \circ s_2$.

DEFINITION 16 (Modified Post's Correspondence Problem)
An *instance $\mathcal{P}$ of PCP* is defined as a finite sequence of pairs of binary words

$$\langle w_1, v_1 \rangle, \cdots, \langle w_{|\mathcal{P}|}, v_{|\mathcal{P}|} \rangle$$

for $|\mathcal{P}| \in \mathbb{N}$ where $|\mathcal{P}| \geq 2$, $w_1 \neq v_1$, and $w_1, v_1 \neq \epsilon$.

A sequence $q_1, \ldots, q_n \in \mathbb{N}$ is called a *solution to* $\mathcal{P}$ if $1 \leq q_i \leq |\mathcal{P}|$ and

$$w_{q_1} w_{q_2} \cdots w_{q_n} w_1 = v_{q_1} v_{q_2} \cdots v_{q_n} v_1 .$$

PCP is a standard example for an undecidable problem.

THEOREM 17
It is undecidable whether an input instance $\mathcal{P}$ of PCP has a solution.

PROOF. See e.g. [22, section 5.2] for a proof. ■

For the rest of the section, we fix an arbitrary instance $\mathcal{P}$ of PCP using the notation from Definition 16. Towards giving $F_{\mathcal{P}}[X, \bar{y}]$, we first distinguish the language the formula will be in.

DEFINITION 18 (Language of $F_{\mathcal{P}}$)
The *language of $F_{\mathcal{P}}$* contains the equality predicate $\approx$ together with the constant symbols

$$0, \epsilon$$

and the unary function symbols

$$\mathsf{s}, \mathsf{s}_0, \mathsf{s}_1, \mathsf{lw}, \mathsf{rw}, \mathsf{p}_1, \ldots, \mathsf{p}_{|\mathcal{P}|}$$

and the unary predicate symbol

$$P,$$

all taken from our given language $\mathcal{L}$. All symbols are assumed to be pairwise different. By $\mathcal{T}$ we denote the set of ground terms in this language.

Regarding the notation of terms, we will often write ft instead of $f(t)$ for f a unary symbol and t a term. Furthermore, for $n \in \mathbb{N}$, f a function symbol, and t a term we define $f^n(t)$ by $f^0(t) = t, f^{n+1}(t) = f(f^n(t))$.

To make the relation between objects on the meta-level and terms in our language clear, we introduce a function $\bar{\cdot}$ defined as follows:

$$\begin{array}{lll} \overline{n} & = \mathsf{s}^n(0) & \text{for } n \in \mathbb{N} \\ \overline{a_1 \cdots a_n} & = \mathsf{s}_{a_1} \cdots \mathsf{s}_{a_n} \epsilon & \text{for a binary word } a_1 \cdots a_n \\ \overline{(q_1, \ldots, q_n)} & = \mathsf{p}_{q_1} \cdots \mathsf{p}_{q_n} 0 & \text{for } q_1, \ldots, q_n \in \{1, \ldots, |\mathcal{P}|\} \end{array}$$

The first part of $F_{\mathcal{P}}$ we will present is the axiomatization of the intended meaning of our language, together with an axiom asserting that $\mathcal{P}$ does not have a solution. To conveniently write down $F_{\mathcal{P}}$, we introduce some notation for formulas.

DEFINITION 19
If Γ, Δ are sets of formulas, the expression $\Gamma \Rightarrow \Delta$ is called a *sequent* and denotes the formula $\bigwedge\Gamma \rightarrow \bigvee\Delta$. For a variable x, if we introduce a set of formulas as $\Gamma[x] = \{F_i[x]\}_{i\in I}$, then for a term t we define $\Gamma[t] := \{F_i[t]\}_{i\in I}$ and for sets of terms T we define $\Gamma[T] := \bigcup_{t\in T}\Gamma[t]$. For a sequent $S[x] = \Gamma[x] \Rightarrow \Delta[x]$, we define $S[t] := \Gamma[t] \Rightarrow \Delta[t]$ and $S[T] = \Gamma[T] \Rightarrow \Delta[T]$. In the context of sequents, set-theoretic union is denoted by comma.

Let $s = a_1 \cdots a_n$ be a binary word and t a term. Then by $s * t$ we denote the term $\mathsf{s}_{a_1} \cdots \mathsf{s}_{a_n} t$. Note that for binary words s_1, s_2 we have $s_1 * \overline{s_2} = \overline{s_1 s_2}$.

DEFINITION 20 (Background Theory)
For a variable x, we define the following sets of formulas:

$$\begin{aligned}
\mathsf{lw\text{-}Ax}[x] := \{ \quad & \mathsf{lw}(\mathsf{p}_1 x) \approx w_1 * (\mathsf{lw}x), \\
& \cdots \\
& \mathsf{lw}(\mathsf{p}_{|\mathcal{P}|} x) \approx w_{|\mathcal{P}|} * (\mathsf{lw}x), \\
& \mathsf{lw}(0) \approx \overline{w_1} \qquad \}.
\end{aligned}$$

$$\begin{aligned}
\mathsf{rw\text{-}Ax}[x] := \{ \quad & \mathsf{rw}(\mathsf{p}_1 x) \approx v_1 * (\mathsf{rw}x), \\
& \cdots \\
& \mathsf{rw}(\mathsf{p}_{|\mathcal{P}|} x) \approx v_{|\mathcal{P}|} * (\mathsf{rw}x), \\
& \mathsf{rw}(0) \approx \overline{v_1} \qquad \}.
\end{aligned}$$

$$\mathsf{ns\text{-}Ax}[x] := \{ \quad \mathsf{lw}(x) \not\approx \mathsf{rw}(x) \quad \}.$$

Before we turn to defining the formula $F_{\mathcal{P}}$ we use to reduce PCP to UBUP, we introduce a notion of standard model for our background theory, and study some properties of standard models. Towards the definition of standard model, we will study the rewrite relation induced by the background theory. We will thus use some basic notions and results on term rewriting systems, see e.g. [3]. Being able to assume that equalities are derived in a directed way will allow us to give simple proofs of properties about equalities which hold in the standard models.

DEFINITION 21
By $\mathcal{R}$ we denote the rewrite relation on $\mathcal{T}$ obtained from the formulas in $\mathsf{lw\text{-}Ax}[x], \mathsf{rw\text{-}Ax}[x]$ by orienting the equations from left to right.

Note that since all function symbols in our language are unary, every $t \in \mathcal{T}$ is of the form $t = f_1 \cdots f_n c$, where the f_i are unary function symbols and $c \in \{0, \epsilon\}$. We define the *size of* t as n. For a strongly normalizing and confluent rewrite relation R on $\mathcal{T}$, we denote by $\sim_R$ the induced equivalence relation on $\mathcal{T}$, and by $[t]_R = \{s \mid t \sim_R s\}$ the equivalence class of a term t.

LEMMA 22
$\mathcal{R}$ is strongly normalizing and confluent. $\mathcal{P}$ has a solution exactly if there exists a term t such that $\mathsf{lw}(t) \sim_{\mathcal{R}} \mathsf{rw}(t)$.

PROOF. By definition of $\mathcal{R}$, we have: if t is not in normal form w.r.t. $\mathcal{R}$, then t contains a term of the form $\mathsf{lw}(t')$ or $\mathsf{rw}(t')$ for some t'. For strong normalization, it therefore suffices to consider only terms of the form $\mathsf{lw}(t), \mathsf{rw}(t)$; induction on the size of t suffices. For confluence, we observe that there are no critical pairs.

Furthermore, by induction on n it is easy to show that $\mathsf{lw}(\overline{(q_1,\ldots,q_n)}) \sim_{\mathcal{R}} w_{q_1}\cdots w_{q_n} w_1$, analogously for rw. Hence $q_1,\ldots,q_n$ is a solution to $\mathcal{P}$ exactly iff $\mathsf{lw}(\overline{(q_1,\ldots,q_n)}) \sim_{\mathcal{R}} \mathsf{rw}(\overline{(q_1,\ldots,q_n)})$. It suffices to observe by induction on the length of a normalizing $\mathcal{R}$-reduction sequence of $\mathsf{lw}(t)$ that if $\mathsf{lw}(t) \sim_{\mathcal{R}} \mathsf{rw}(t)$, then $t = \overline{(q_1,\ldots,q_n)}$ for some $q_1,\ldots,q_n \in \mathbb{N}$. ■

Based on $\mathcal{R}$, we can now define our notion of standard model.

DEFINITION 23 (Standard models)
Let $\mathcal{M} = (M, I)$ be a structure. $\mathcal{M}$ is called a *standard model* if $M = \{[t]_{\mathcal{R}} \mid t \in \mathcal{T}\}$ and for all $t \in \mathcal{T}$, $I(t) = [t]_{\mathcal{R}}$. A formula F is called *standard valid* if it holds in all standard models.

LEMMA 24
The formulas $\forall x \bigwedge \mathsf{lw\text{-}Ax}[x]$ and $\forall x \bigwedge \mathsf{rw\text{-}Ax}[x]$ are standard valid. The formula $\forall x\, \mathsf{ns\text{-}Ax}[x]$ is standard valid exactly if $\mathcal{P}$ has no solution.

PROOF. The first part is immediate by definition, the second part by Lemma 22. ■

It is simple to construct standard models with an arbitrary interpretation of the P predicate symbol.

LEMMA 25
For all $N \subseteq \{[t]_{\mathcal{R}} \mid t \in \mathcal{T}\}$ there exists a standard model (M, I) such that $I(P) = N$.

If S is a set of predicate symbols, then a formula in the language of $F_{\mathcal{P}}$ containing only predicate symbols from S is called an *S-formula*. We introduce some standard validity preserving transformations on formulas. In the following result, $\top$ denotes a fixed valid $\{P\}$-formula (e.g. $P(0) \vee \neg P(0)$), and a formula F is called *standard unsatisfiable* if it holds in no standard model.

LEMMA 26
Let F be a ground quantifier-free formula.

(1) If F is a $\{\approx, P\}$-formula, and F' is the formula obtained from F by replacing standard valid $\approx$-atoms by $\top$, and standard unsatisfiable $\approx$-atoms by $\neg\top$, then $F \leftrightarrow F'$ is standard valid and F' is a $\{P\}$-formula.
(2) If F is a standard valid $\{P\}$-formula, s, t terms, and F' the formula obtained from F by replacing all terms t' with $t' \sim_{\mathcal{R}} t$ by s, then F' is standard valid.

PROOF. Note that for any standard model $\mathcal{M}$, if $t, s \in \mathcal{T}$ then $\mathcal{M} \models s \approx t$ exactly if $s \sim_{\mathcal{R}} t$, and $\mathcal{M} \models s \not\approx t$ exactly if $s \not\sim_{\mathcal{R}} t$. Hence every ground $\approx$-atom is either standard valid or standard unsatisfiable, yielding item 1. For item 2, let $(M, I) \not\models F'$. Note that if a term u becomes u' by the replacement, then there is a unique term $r_u[x]$ and a unique term v_u s.t. $u = r_u[v_u]$ with $v_u \sim_{\mathcal{R}} t$, and $u' = r_u[s]$. Hence the set S defined by

$$[r_u[v_u]]_{\mathcal{R}} \in S :\Leftrightarrow [r_u[s]]_{\mathcal{R}} \in I(P)$$

is well-defined, and the model $\mathcal{N}$ obtained from S by Lemma 25 fulfills $\mathcal{N} \not\models F$. ■

LEMMA 27
Let $s[x], t[x]$ be terms containing x and $d, n, m \in \mathbb{N}$ with $m, n > 0$.

(1) If $s[\overline{n}] \approx q$ is standard valid for some term q, then q contains $\overline{n}$.
(2) $s[\overline{n}] \approx t[\overline{m}] \leftrightarrow s[\overline{n+d}] \approx t[\overline{m+d}]$ is standard valid.

PROOF. By inspection of $\mathcal{R}$, we note that if $t \sim_{\mathcal{R}} t'$ then t contains $\overline{n}$ iff t' contains $\overline{n}$. This yields item 1. For item 2, note that if t rewrites to t' via $\mathcal{R}$, then the redex is not a numeral. Hence we can

simulate the rewrite sequences which reduce $s[\overline{n}], t[\overline{m}]$ to the same normal form on $s[\overline{n+d}], t[\overline{m+d}]$ to reduce them to the same normal form. ■

This concludes our study of standard models. We now turn to performing the reduction of PCP to UBUP.

DEFINITION 28 (Formula $F_{\mathcal{P}}$)
We define the sequents

$$S_1[X,\beta] := \mathsf{lw\text{-}Ax}[\beta], \mathsf{rw\text{-}Ax}[\beta], \mathsf{ns\text{-}Ax}[\beta], P(0) \Rightarrow X(0,\beta)$$

$$S_2[X,\gamma,\nu] := \begin{cases} \mathsf{lw\text{-}Ax}[\gamma], \mathsf{rw\text{-}Ax}[\gamma], \mathsf{ns\text{-}Ax}[\gamma], P(\gamma) \rightarrow P(\mathsf{s}\gamma), \\ X(\nu,\gamma), X(\nu,\mathsf{p}_1(\gamma)), \cdots, X(\nu,\mathsf{p}_{|\mathcal{P}|}(\gamma)), X(\nu,\mathsf{s}\gamma) \Rightarrow \\ X(\mathsf{s}\nu,\gamma) \end{cases}$$

$$S_3[X,\alpha] := X(\alpha,0) \Rightarrow P(\alpha)$$

Finally, we define

$$F_{\mathcal{P}}[X,\alpha,\beta,\gamma,\nu] := S_1[X,\beta] \wedge S_2[X,\gamma,\nu] \wedge S_3[X,\alpha].$$

The intuitive meaning of $X(\overline{n},\overline{s})$, for $n \in \mathbb{N}$ and $s = (q_1,\ldots,q_k) \in \mathbb{N}^k$, is 'there are no $q_{k+1},\ldots,q_{k+n}$ such that $(q_1,\ldots,q_{k+n})$ is a solution of $\mathcal{P}$', and the intuitive meaning of $P(\overline{n})$ is 'there is no solution of $\mathcal{P}$ of size n'. Our first main task will be to show the following.

LEMMA 29
If $\forall\alpha\beta\gamma\nu F_{\mathcal{P}}[X,\alpha,\beta,\gamma,\nu]$ is a positive instance of UBUP, then $\mathcal{P}$ has a solution.

The strategy for proving this lemma is to assume that $\mathcal{P}$ has no solution, and to argue for contradiction by using the validity of $F_{\mathcal{P}}$ to derive validity of (roughly) $P(\overline{q})$ for some $q \in \mathbb{N}$ which is large w.r.t. the witness of $F_{\mathcal{P}}$, contradicting Lemma 25. We start by studying some standard valid sequents.

DEFINITION 30 (Sets $\mathcal{C}_n(t)$, sequents Θ_n, Σ_n)
For $n \in \mathbb{N}$ and a term t we define the set of terms

$$\mathcal{C}_n(t) := \{f_1 \cdots f_q t \mid 0 \leq q \leq n, f_i \in \{\mathsf{s},\mathsf{p}_1,\ldots,\mathsf{p}_{|\mathcal{P}|}\}\}.$$

Furthermore, for a formula $G[y,z]$ and variables γ,ν, we define the sequents

$$\Theta_n[G,\gamma,\nu] := \{P(t) \rightarrow P(\mathsf{s}(t)) \mid t \in \mathcal{C}_{n-1}(\gamma)\}, G[\nu,\mathcal{C}_n(\gamma)] \Rightarrow G[\mathsf{s}^n\nu,\gamma]$$
$$\Sigma_n[G,\gamma] := P(0), \{P(t) \rightarrow P(\mathsf{s}(t)) \mid t \in \mathcal{C}_{n-1}(\gamma)\} \Rightarrow G[\mathsf{s}^n 0,\gamma]$$

where $\mathcal{C}_{-1}(t) := \emptyset$.

Note that for the second sequent of the definition of $F_{\mathcal{P}}$ we have

$$S_2[X,\gamma,\nu] = \begin{cases} \mathsf{lw\text{-}Ax}[\gamma], \mathsf{rw\text{-}Ax}[\gamma], \mathsf{ns\text{-}Ax}[\gamma], P(\gamma) \rightarrow P(\mathsf{s}\gamma), \\ \{X(\nu,t) \mid t \in \mathcal{C}_1(\gamma)\} \Rightarrow X(\mathsf{s}\nu,\gamma) \end{cases}.$$

When reasoning with sequents, we will apply the following well-known inference rule called *cut*.

LEMMA 31 (Cut rule)
Let $\mathcal{M}$ be a structure, $\Gamma,\Delta,\Pi,\Lambda$ sets of formulas, and C a formula. Then $\mathcal{M} \models \Gamma \Rightarrow \Delta, C$ and $\mathcal{M} \models C, \Pi \Rightarrow \Lambda$ imply that $\mathcal{M} \models \Gamma, \Pi \Rightarrow \Delta, \Lambda$.

LEMMA 32
Assume that $\mathcal{P}$ has no solution and that $G[y,z]$ is a witness of $\forall\alpha\beta\gamma\nu F_{\mathcal{P}}[X,\alpha,\beta,\gamma,\nu]$. Then, for all $n\in\mathbb{N}$, the formulas $\Theta_n[G,\gamma,\nu]$ and $\Sigma_n[G,\gamma]$ are standard valid.

PROOF. By assumption, $S_2[G,\gamma,\nu]$ is valid, hence for the sequent

$$S_2'[G,\gamma,\nu]:=P(\gamma)\to P(\mathsf{s}\gamma),G[\nu,\mathcal{C}_1(\gamma)]\Rightarrow G[\mathsf{s}\nu,\gamma]$$

we have that $S_2'[G,\gamma,\nu]$ is standard valid by Lemma 24. Hence for all $k\in\{1,\dots,n-1\}$, the sequent

$$\mathcal{S}_k:=S_2'[G,\mathcal{C}_k(\gamma),\mathsf{s}^{n-k-1}\nu]$$

as well as the sequent $T:=S_2'[G,\gamma,\mathsf{s}^{n-1}\nu]$ is standard valid. Since $\mathsf{p}_1\gamma,\dots,\mathsf{p}_{|\mathcal{P}|}\gamma,\mathsf{s}\gamma\in\mathcal{C}_1(\gamma)$ and $\mathcal{C}_0(\gamma)\subseteq\mathcal{C}_1(\gamma)$, we can cut $\mathcal{S}_1$ with T to show that the sequent

$$\{P(t)\to P(\mathsf{s}t)\,|\,t\in\mathcal{C}_1(\gamma)\},G[\mathsf{s}^{n-2}\nu,\mathcal{C}_2(\gamma)]\Rightarrow G[\mathsf{s}^n\nu,\gamma]$$

is standard valid. We continue cutting with $\mathcal{S}_2,\dots,\mathcal{S}_{n-1}$ until we eventually obtain that $\Theta_n[G,\gamma,\nu]$ is standard valid.

Furthermore, by assumption and Lemma 24, the sequent

$$S_1'[G,\gamma]:=P(0)\Rightarrow G[0,\gamma]$$

is standard valid. Cutting $S_1'[G,t]$, for all $t\in\mathcal{C}_1(\gamma)$, with $S_2'[G,\gamma,0]$ yields standard validity of

$$P(0),\{P(t)\to P(\mathsf{s}(t))\,|\,t\in\mathcal{C}_0(\gamma)\}\Rightarrow G[\mathsf{s}0,\gamma].$$

Cutting this sequent under the substitution $[\gamma\backslash t]$, for $t\in\mathcal{C}_1(\gamma)$, with the sequent $S_2'[G,\gamma,\mathsf{s}0]$ yields standard validity of

$$P(0),\{P(t)\to P(\mathsf{s}(t))\,|\,t\in\mathcal{C}_1(\gamma)\}\Rightarrow G[\mathsf{s}^20,\gamma].$$

Continuing inductively yields standard validity of $\Sigma_n[G,\gamma]$. ■

We are now ready to prove our first main result.

PROOF OF LEMMA 29. Assume that $\mathcal{P}$ has no solution. If $F_{\mathcal{P}}$ is a positive instance of UBUP, there is a quantifier-free formula $G[y,z]$ such that $\forall\alpha\beta\gamma\nu F_{\mathcal{P}}[G,\alpha,\beta,\gamma,\nu]$ is valid. Let m be an upper bound on the sizes of the terms in $G[y,z]$. We choose $q_1,q_2,n\in\mathbb{N}$ with $q_2>q_1>2n$ and $n>m$ which ensures not only $q_2>q_1$ but also that q_1 and q_2 are sufficiently large to allow replacing $\overline{q_1}$ by $\overline{q_2}$ without destroying standard validity of a formula we are going to construct now. By Lemma 32, the ground formula

$$\Theta_n[G,0,\overline{q_1-n}]=\{P(t)\to P(\mathsf{s}(t))\,|\,t\in\mathcal{C}_{n-1}(0)\},G[\overline{q_1-n},\mathcal{C}_n(0)]\Rightarrow G[\overline{q_1},0]$$

is standard valid. By Lemma 26.1, for all $t,s\in\mathcal{T}$ we can associate to $G[s,t]$ a quantifier-free ground $\{P\}$-formula $H_{s,t}$ such that $G[s,t]\leftrightarrow H_{s,t}$ is standard valid. Therefore the $\{P\}$-formula

$$\{P(t)\to P(\mathsf{s}(t))\,|\,t\in\mathcal{C}_{n-1}(0)\},\{H_{\overline{q_1-n},t}\,|\,t\in\mathcal{C}_n(0)\}\Rightarrow H_{\overline{q_1},0}$$

is standard valid. Replacing the term $\overline{q_1}$ by $\overline{q_2}$ in this formula yields the formula

$$T:=\{P(t)\to P(\mathsf{s}(t))\,|\,t\in\mathcal{C}_{n-1}(0)\},\{H_{\overline{q_1-n},t}\,|\,t\in\mathcal{C}_n(0)\}\Rightarrow H_{\overline{q_1},0}[\overline{q_1}\backslash\overline{q_2}]$$

since $\overline{q_1}$ cannot occur on the left-hand side. This sequent is standard valid by Lemma 26.2. We have $H_{\overline{q_1},0}[\overline{q_1}\backslash\overline{q_2}]=H_{\overline{q_2},0}$ since, by Lemma 27, the atoms $t[\overline{q_1}]\approx s[\overline{q_1}]$ in $G[\overline{q_1},0]$ are standard

valid (standard unsatisfiable) iff the corresponding atoms $t[\overline{q_2}] \approx s[\overline{q_2}]$ in $G[\overline{q_2}, 0]$ are standard valid (standard unsatisfiable).

Furthermore, by Lemma 32 for any $t \in \mathcal{C}_n(0)$ the sequent

$$\Sigma_n[G,t] = P(0), \{P(u) \to P(\mathsf{s}u) \mid u \in \mathcal{C}_{q_1-n-1}(t)\} \Rightarrow G[\overline{q_1 - n}, t]$$

is standard valid. Replacing $G[\overline{q_1 - n}, t]$ by $H_{q_1-n,t}$ in these sequents and cutting with T, we obtain standard validity of

$$P(0), \{P(u) \to P(\mathsf{s}u) \mid u \in \mathcal{C}_{q_1-1}(0)\} \Rightarrow H_{\overline{q_2},0}$$

which implies, using validity of $S_3[G, \overline{q_2}]$, the standard validity of

$$P(0), \{P(u) \to P(\mathsf{s}u) \mid u \in \mathcal{C}_{q_1-1}(0)\} \Rightarrow P(\overline{q_2}).$$

Define $N := \{[\overline{k}]_{\mathcal{R}} \mid k < q_2\}$, then the standard model obtained by Lemma 25 is a countermodel to the latter sequent, contradicting standard validity. ■

We now turn our attention to the second direction of our main result.

LEMMA 33
If $\mathcal{P}$ has a solution, then $\forall \alpha \beta \gamma \nu F_{\mathcal{P}}[X, \alpha, \beta, \gamma, \nu]$ is a positive instance of UBUP.

The proof strategy for this Lemma is as follows: remember that the intuitive meaning of the witness $G[\overline{n}, \overline{s}]$ of $F_{\mathcal{P}}$ was that 'there is no solution of $\mathcal{P}$ that can be obtained from s by extending it by n numbers'. Given a solution of $\mathcal{P}$, we will define the formula $G[y,z]$ asserting this by a finite case distinction—since there exists a solution, $G[\overline{n}, 0]$ for large enough n will be unsatisfiable, making each sequent of $F_{\mathcal{P}}$ valid. For the proof, we will need some sets of formulas.

DEFINITION 34 (Sets $\mathsf{L}_n, \mathsf{R}_n, \mathsf{N}_n$, formula W_n)
For $n \in \mathbb{N}$ we define the sets of formulas

$$\mathsf{L}_n := \mathsf{lw\text{-}Ax}[\mathcal{C}_n(0)], \qquad \mathsf{R}_n := \mathsf{rw\text{-}Ax}[\mathcal{C}_n(0)], \qquad \mathsf{N}_n := \mathsf{ns\text{-}Ax}[\mathcal{C}_n(0)].$$

For a variable z, we furthermore define the formula $W_n[z]$ by

$$W_0[z] := P(0) \wedge \mathsf{lw\text{-}Ax}[z] \wedge \mathsf{rw\text{-}Ax}[z] \wedge \mathsf{ns\text{-}Ax}[z]$$
$$W_{n+1}[z] := (P(z) \to P(\mathsf{s}z)) \wedge \bigwedge W_n[\mathcal{C}_1(z)]$$

The following properties of $W_n[z]$ follow easily from the definition.

LEMMA 35
If $m > n$ then $W_m[z] \to W_n[z]$ is valid. For all $n \in \mathbb{N}$, $W_n[0] \to P(\overline{n})$ is valid. For all $n \in \mathbb{N}$ and $F \in \mathsf{L}_n \cup \mathsf{R}_n \cup \mathsf{N}_n$, $W_n[0] \to F$ is valid.

LEMMA 36
If $q_1, \ldots, q_n$ is a solution of $\mathcal{P}$, then $\mathsf{L}_n, \mathsf{R}_n, \mathsf{N}_n \Rightarrow$ is valid.

PROOF. From $\mathsf{L}_n \cup \mathsf{R}_n \cup \mathsf{N}_n$ we can derive $\mathsf{lw}(\overline{(q_1, \ldots, q_n)}) \approx w_{q_1} \cdots w_{q_n} w_1$ and $\mathsf{rw}(\overline{(q_1, \ldots, q_n)}) \approx v_{q_1} \cdots v_{q_n} v_1$. Since $q_1, \ldots, q_n$ is a solution, we can derive $\mathsf{lw}(\overline{(q_1, \ldots, q_n)}) \approx \mathsf{rw}(\overline{(q_1, \ldots, q_n)})$, yielding a contradiction with N_n since $\overline{(q_1, \ldots, q_n)} \in \mathcal{C}_n(0)$. ■

LEMMA 37
If $q_1, \ldots, q_n$ is a solution of $\mathcal{P}$, then $\neg W_n[0]$ is valid.

PROOF. By Lemmas 35 and 36. ■

We can now finish our proof.

PROOF OF LEMMA 33. Assume that $i_1, i_2, \cdots, i_\ell$ is a solution of $\mathcal{P}$, and define the formula $G[y,z]$ as follows.

$$\begin{array}{ll}
y \approx 0 \rightarrow & W_0[z] \wedge \\
y \not\approx 0 \wedge y \approx \overline{1} \rightarrow & W_1[z] \wedge \\
\cdots & \\
y \not\approx 0 \wedge y \not\approx \overline{1} \wedge \cdots \wedge y \approx \overline{\ell-1} \rightarrow & W_{\ell-1}[z] \wedge \\
y \not\approx 0 \wedge y \not\approx \overline{1} \wedge \cdots \wedge y \not\approx \overline{\ell-1} \rightarrow & W_\ell[z]
\end{array}$$

We claim that $G[y,z]$ is a witness of $F_{\mathcal{P}}$, i.e. that $\forall\alpha\beta\gamma\nu F_{\mathcal{P}}[G,\alpha,\beta,\gamma,\nu]$ is valid. It suffices to show that each of the formulas $\forall\beta S_1[G,\beta]$, $\forall\gamma\nu S_2[G,\gamma,\nu]$, and $\forall\alpha S_3[G,\alpha]$ are valid. For $\forall\beta S_1[G,\beta]$, this is easy to see. For the second sequent, we have to prove $G[\mathsf{s}\nu,\gamma]$ from assumptions

$$\mathsf{lw\text{-}Ax}[\gamma], \mathsf{rw\text{-}Ax}[\gamma], \mathsf{ns\text{-}Ax}[\gamma], P(\gamma) \rightarrow P(\mathsf{s}\gamma), \{G[\nu,t] \mid t \in \mathcal{C}_1(\gamma)\}.$$

We sketch a formal proof. By classical logic, we distinguish:

(1) $\mathsf{s}\nu \approx 0$. Then $G[\mathsf{s}\nu,\gamma]$ is equivalent to $W_0[\gamma]$. By classical logic, we distinguish:
   (a) $\nu \approx 0$. Then $G[\nu,\gamma]$ is equivalent to $W_0[\gamma]$, finishing the proof.
   (b) $\nu \approx \overline{m}$ and $\nu \not\approx \overline{0}, \ldots, \nu \not\approx \overline{k-1}$ for some $1 \leq m < \ell$. Then $G[\nu,\gamma]$ is equivalent to $W_m[\gamma]$ which implies $W_0[\gamma]$ by Lemma 35.
   (c) $\nu \not\approx \overline{0}$, …, $\nu \not\approx \overline{\ell-1}$. Then $G[\nu,\gamma]$ is equivalent to $W_\ell[\gamma]$, which suffices as in the previous case.

(2) $\mathsf{s}\nu \approx \overline{k}$ and $\mathsf{s}\nu \not\approx \overline{0}, \ldots, \overline{k-1}$ for some $1 \leq k < \ell$. We derive $\nu \not\approx \overline{k-2}, \nu \not\approx \overline{k-3} \ldots \nu \not\approx \overline{0}$ from compatibility and reflexivity of $\approx$. For every $t \in \mathcal{C}_1(\gamma)$, from these equations and $G[\nu,t]$ we derive

$$\begin{array}{ll}
\nu \approx \overline{k-1} \rightarrow & W_{k-1}[t] \wedge \\
\nu \not\approx \overline{k-1} \wedge \nu \approx \overline{k} \rightarrow & W_k[t] \wedge \\
\cdots & \\
\nu \not\approx \overline{k-1} \wedge \nu \not\approx \overline{k} \wedge \cdots \wedge \nu \approx \overline{\ell-1} \rightarrow & W_{\ell-1}[t] \wedge \\
\nu \not\approx \overline{k-1} \wedge \nu \not\approx \overline{k} \wedge \cdots \wedge \nu \not\approx \overline{\ell-1} \rightarrow & W_\ell[t]
\end{array}$$

which implies

$$\Rightarrow W_{k-1}[t], \ldots, W_\ell[t]$$

which in turn implies $W_{k-1}[t]$ by Lemma 35. Together with $P(\gamma) \rightarrow P(\mathsf{s}\gamma)$ we derive

$$P(\gamma) \rightarrow P(\mathsf{s}\gamma) \wedge \bigwedge W_{k-1}[\mathcal{C}_1(\gamma)]$$

which is exactly $W_k[\gamma]$. By Lemma 35 we thus have $W_i[\gamma]$ for $0 \leq i \leq k$, and hence the first $k+1$ implications in $G[s\nu,\gamma]$ hold. In the other $\ell - k$ implications, $s\nu \not\approx \overline{k}$ occurs in the antecedent, hence they follow trivially from our assumption. In total, this yields $G[s\nu,\gamma]$.

(3) $\mathsf{s}\nu \not\approx \overline{0}, \ldots, \mathsf{s}\nu \not\approx \overline{\ell-1}$. We proceed analogously to the previous case.

It remains to give a formal proof of the third sequent, i.e. a proof of $P(\alpha)$ from the assumption $G[\alpha,0]$. By classical logic, we distinguish:

(1) $\alpha \approx 0$. $G[\alpha,0]$ yields $W_0[0]$ which yields $P(0)$.
(2) $\alpha \approx \overline{k}$ for some $1 \leq k < \ell$, and $\alpha \not\approx \overline{0}, \ldots, \overline{k-1}$. $G[\alpha,0]$ yields $W_k[0]$ which implies $P(\overline{k})$ by Lemma 35.
(3) $\alpha \not\approx \overline{0}, \ldots, \alpha \not\approx \overline{\ell}$. $G[\alpha,0]$ yields $W_\ell[0]$. But Lemma 37 yields $\neg W_\ell[0]$, and we may derive $P(\alpha)$.■

Having proven Lemmas 29 and 33 we obtain the main result of this section.

THEOREM 38
The UBUP problem is undecidable.

PROOF. It suffices to note that $F_{\mathcal{P}}$ can be computed from $\mathcal{P}$ by a Turing machine. ■

7 UBUP in automated theorem proving

As stated in the introduction, instances of UBUP naturally occur in the context of algorithms for automated theorem proving and proof compression introduced in [7, 13–15].

In [13–15], algorithms are presented which compress analytic proofs by introducing Π_1-cuts. A step of the presented algorithm for the case of one cut is the solution of instances of UBUP in order to obtain the cut-formula (see e.g. [14], section 3.3). In [15], it could be proved that the mentioned instances of UBUP are always solvable. In [14], some heuristics are defined to find solutions of the mentioned UBUP instances which have a small logical complexity.

In [7], an algorithm for automated inductive theorem proving is presented. As stated in section 1, the undecidability of UBUP is relevant for this work: the algorithm IndProof presented there automatically generates inductive invariants which are in many cases useful to prove a given universal statement. Analogously to the papers [13–15] where an UBUP problem has to be solved to find suitable cut-formulas, in [7] such a problem has to be solved to find a suitable induction formula. IndProof attacks the mentioned UBUP problem using heuristics without giving a guarantee to always find a witness if one exists. Of course, the undecidability result for UBUP proved in this article does not justify the use of heuristics instead of a complete algorithm since the instances of UBUP actually occurring during the computations of IndProof might be decidable as it is the case for the algorithms presented in the papers [13–15].

Nevertheless, using the same techniques as in the present article, the undecidability result for UBUP can be slightly strengthened to show that already the restriction of UBUP to those instances occuring in computations of IndProof is undecidable. The strengthened result justifies the above mentioned use of heuristics in the search of a witness of UBUP instances.

8 Conclusion

The problem of BUP as defined on page 110 has been fully characterized in our article:

- BUP restricted to quantifier-free formulas (QFBUP) is Π_2^P-complete.
- BUP restricted to Π_1 (UBUP) or Σ_1 (EBUP) first-order formulas is undecidable.

Nevertheless, some interesting closely related questions remain open:

- In most parts of the present article, we work in first-order logic with equality. Note that the arguments in sections 5.1 and 6 heavily rely on the use of equations. Therefore, the complexity of restrictions of BUP in the setting of first-order logic without equality remains open.
- It remains unclear how complicated the underlying logical languages $\mathcal{L}$ have to be to make UBUP restricted to $\mathcal{L}$-formulas $F[X]$ undecidable. For example, the question whether undecidability already holds for X restricted to predicate variables of arity one remains open.

Acknowledgements

The authors would like to thank the anonymous referees of this article: their suggestion to use the DLS algorithm to prove Theorem 2, as well as their suggestion to prove Π_2^P-membership of QFBUP via reduction to Π_2-TQBF, greatly improved the presentation of the results of this article.

Funding

This work was supported by the Vienna Science and Technology Fund (WWTF) as part of the Vienna Research Group 12-04.

References

[1] W. Ackermann. Untersuchungen über das Eliminationsproblem der mathematischen Logik. *Mathematische Annalen*, 390–413, 1935.
[2] F. Baader. On the complexity of boolean unification. *Information Processing Letters*, **67**, 215–220, 1998.
[3] F. Baader and T. Nipkow. *Term Rewriting and All That*. Cambridge University Press, 1998.
[4] F. Baader and W. Snyder. Unification theory. In *Handbook of Automated Reasoning*, A. Robinson and A. Voronkov, eds, pp. 445–532, 2001.
[5] W. Büttner and H. Simonis. Embedding boolean exprerssions into logic programming. *Journal of Symbolic Computation*, **4**, 191–205, 1987.
[6] P. Doherty, W. Łukaszewicz and A. Szałas. Computing circumscription revisited: a reduction algorithm. *Journal of Automated Reasoning*, **18**, 297–336, 1997.
[7] S. Eberhard and S. Hetzl. Inductive theorem proving based on tree grammars. *Annals of Pure and Applied Logic*, **166**, 665–700, 2015.
[8] H. Friedman. The disjunction property implies the numerical existence property. In *Proceedings of the National Academy of Sciences of the United states of America*, **72**, 2877–2878, 1975.
[9] D. M. Gabbay and H. J. Ohlbach. Quantifier elimination in second–order predicate logic. In *Principles of Knowledge Representation and Reasoning (KR92)*, B. Nebel, C. Rich and W. Swartout, eds, pp. 425–435. Morgan Kaufmann, 1992.
[10] D. M. Gabbay, R. A. Schmidt and A. Szałas. *Second-Order Quantifier Elimination*. College Publications, 2008.
[11] A. Gascón, G. Godoy, M. Schmidt-Schauß and A. Tiwari. Context unification with one context variable. *Journal of Symbolic Computation*, **45**, 173–193, 2010.
[12] W. D. Goldfarb. The undecidability of the second-order unification problem. *Theoretical Computer Science*, **13**, 225–230, 1981.
[13] S. Hetzl, A. Leitsch, G. Reis, J. Tapolczai and D. Weller. Introducing quantified cuts in logic with equality. In *IJCAR*, Vol. 8562 of *Lecture Notes in Computer Science*, pp. 240–254, 2014.

[14] S. Hetzl, A. Leitsch, G. Reis and D. Weller. Algorithmic introduction of quantified cuts. *Theoretical Computer Science*, **549**, 1–16, 2014.
[15] S. Hetzl, A. Leitsch and D. Weller. Towards Algorithmic Cut-Introduction. In *Logic for Programming, Artificial Intelligence and Reasoning (LPAR-18)*, Vol. 7180 of *Lecture Notes in Computer Science*, pp. 228–242. Springer, 2012.
[16] U. Martin and T. Nipkow. Boolean unification - the story so far. *Journal of Symbolic Computation*, **7**, 275–293, 1989.
[17] U. Martin and T. Nipkow. Unification in boolean rings. *Journal of Automated Reasoning*, **4**, 381–396, 1989.
[18] C. H. Papadimitriou. *Computational Complexity*. Addison-Wesley, 1994.
[19] J. A. Robinson. A machine oriented logic based on the resolution principle. *Journal of the ACM*, **10**, 163–174, 1965.
[20] M. Schmidt-Schauß. A decision algorithm for stratified context unification. *Journal of Logic and Computation*, **12**, 929–953, 2002.
[21] M. Schmidt-Schauß and K. U. Schulz. Solvability of context equations with two context variables is decidable. *Journal of Symbolic Computation*, **33**, 77–122, 2002.
[22] M. Sipser. *Introduction to the Theory of Computation*, 2nd edn., Thomson, 2006.

Received 20 June 2014