An Algorithm for Existential Boolean Unification with Predicates (Extended Abstract)

Fabian Achammer[1,*], Stefan Hetzl[1]

[1]TU Wien, Austria

Abstract

We present a first draft of an algorithm to solve existential Boolean unification with predicates. The algorithm is based on Herbrand's theorem, the witness construction for quantifier-free Boolean unification and the formula instantiation problem.

Keywords

Boolean unification with Predicates, Formula instantiation, Herbrand skeletons

1. Introduction

In this paper we consider the problem of *Boolean Unification with Predicates (BUP)*: Given a first-order formula $\varphi(\overline{X})$ with free predicate variables $\overline{X} = (X_1, \ldots, X_n)$ such that X_i has arity k_i, find quantifier-free predicate expressions $\alpha_1, \ldots, \alpha_n$ such that α_i has arity k_i and $\varphi[X_1 \leftarrow \alpha_1, \ldots, X_n \leftarrow \alpha_n]$ is valid in first-order logic with equality. Such α_i are called *witnesses* for φ.

For example, the formula $X(a) \wedge X(b)$ has a witness given by $\alpha = \lambda u.u \simeq a \vee u \simeq b$, since $\alpha(a) \wedge \alpha(b) = (a \simeq a \vee a \simeq b) \wedge (b \simeq a \vee b \simeq b)$ is valid in first-order logic with equality. On the other hand, the formula $X(a) \wedge \neg X(b)$ does not have a witness, since the formula is not satisfied by any model that equates a and b.

BUP is recursively enumerable, but undecidable since it contains the first-order validity problem. The following subclasses of BUP have been studied in [1]:

1. Quantifier-free Boolean unification with predicates (QFBUP): For a given quantifier-free formula $\varphi(\overline{X})$, find a quantifier-free witness.
2. Existential Boolean unification with predicates (EBUP): For a given formula $\exists \overline{u}\, \varphi(\overline{X}, \overline{u})$, where $\varphi(\overline{X}, \overline{u})$ is quantifier-free, find a quantifier-free witness.
3. Universal Boolean unification with predicates (UBUP): For a given formula $\forall \overline{u}\, \varphi(\overline{X}, \overline{u})$, where $\varphi(\overline{X}, \overline{u})$ is quantifier-free, find a quantifier-free witness.

QFBUP is decidable while EBUP and UBUP are both undecidable [1]. Still, all problems are recursively enumerable, but besides the trivial enumeration algorithm, no semi-decision procedures are known for EBUP or UBUP. In particular, no practical algorithm, which would be suitable for implementation, is known.

In this paper we present a first draft of a sound and complete algorithm for EBUP and illustrate it on an example. We present the algorithm for a single predicate variable X. The generalization to an arbitrary finite amount of predicate variables is straightforward. The algorithm is based on Herbrand's theorem, the witness construction in [1] for QFBUP and solving the formula instantiation problem [2]: Given a quantifier-free first-order formula φ find a first-order substitution σ such that $\varphi\sigma$ is valid in first-order logic without equality. The idea is to use Herbrand's theorem and formula instantiation to

CI-BD-SOQE 2026: Workshop on Craig Interpolation, Beth Definability, and Second-Order Quantifier Elimination, at FLoC 2026: The 9th Federated Logic Conference, July 24–25, 2026, Lisbon, Portugal

*Corresponding author.

fabian.achammer@tuwien.ac.at (F. Achammer); stefan.hetzl@tuwien.ac.at (S. Hetzl)

https://fabian.achammer.dev/ (F. Achammer); https://dmg.tuwien.ac.at/hetzl/ (S. Hetzl)

0009-0002-3799-6393 (F. Achammer); 0000-0002-6461-5982 (S. Hetzl)

take care of the first-order existential quantifiers and combine this with the algorithm for QFBUP to find solutions for the second-order variables.

More precisely the algorithm proceeds like this: Consider an input formula $\exists \overline{u}\, \varphi(X, \overline{u})$ where $\varphi(X, \overline{u})$ is quantifier-free and X is a predicate variable. For $n \in \mathbb{N}$ the n-fold Herbrand skeleton

$$\varphi^n(X, \overline{u_1}, \ldots, \overline{u_n}) := \varphi(X, \overline{u_1}) \vee \cdots \vee \varphi(X, \overline{u_n})$$

is a quantifier-free formula. By applying the QFBUP-algorithm to $\varphi^n(X, \overline{u_1}, \ldots, \overline{u_n})$ we get a quantifier-free witness $\alpha(\overline{u_1}, \ldots, \overline{u_n})$ for $\varphi^n(X, \overline{u_1}, \ldots, \overline{u_n})$[1] which can contain the free variables $\overline{u_1}, \ldots, \overline{u_n}$. Then we substitute this witness into the Herbrand skeleton to get a quantifier-free formula

$$\psi_n(\overline{u_1}, \ldots, \overline{u_n}) := \varphi^n(\overline{u_1}, \ldots, \overline{u_n})[X \leftarrow \alpha(\overline{u_1}, \ldots, \overline{u_n})]$$

which only contains the free first-order variables $\overline{u_1}, \ldots, \overline{u_n}$. The goal is now to find terms $\overline{t_1}, \ldots, \overline{t_n}$ such that $\psi_n(\overline{t_1}, \ldots, \overline{t_n})$ is valid in which case $\alpha(\overline{t_1}, \ldots, \overline{t_n})$ is the witness we were looking for. This means solving the formula instantiation problem for the quantifier-free formula $\psi_n(\overline{u_1}, \ldots, \overline{u_n})$. In first-order logic with equality this problem is undecidable [3, 4]. However, in first-order logic without equality this problem is decidable so we instead axiomatize equality as a first-order universal theory $T_{\mathrm{eq}} = \forall \overline{v}\, \psi_{\mathrm{eq}}(\overline{v})$ where $\psi_{\mathrm{eq}}(\overline{v})$ is quantifier-free. We then apply Herbrand's theorem to $\forall \overline{v}\, \psi_{\mathrm{eq}}(\overline{v}) \rightarrow \psi_n(\overline{t_1}, \ldots, \overline{t_n})$ in first-order logic without equality to instead solve the problem of, given $k \in \mathbb{N}$ finding $\overline{t_1}, \ldots, \overline{t_n}$ and $\overline{s_1}, \ldots, \overline{s_k}$ such that

$$\psi_{\mathrm{eq}}(\overline{s_1}) \wedge \cdots \wedge \psi_{\mathrm{eq}}(\overline{s_k}) \rightarrow \psi_n(\overline{t_1}, \ldots, \overline{t_n}) \tag{$*$}$$

is valid in first-order logic without equality. This way, we get around the undecidability in the equality case by instead parameterizing the amount of instances of equality axioms used. This algorithm improves on the naive enumeration of EBUP-solutions by instead enumerating all $(n, k) \in \mathbb{N}^2$, using the QFBUP-algorithm to find witnesses for the predicate variable and solving instances of formula instantiation.

Throughout this paper we denote by $\models$ the satisfiability relation in first-order logic *without* equality and by $\models_{\mathrm{eq}}$ the satisfiability relation in first-order logic *with* equality. Furthermore we denote equality on the object level by the symbol $\simeq$.

The structure of this paper is as follows: In Section 2 we recall the QFBUP method of constructing witnesses which is based on the DLS algorithm. Section 3 deals with the formula instantiation problem. In Section 4 we put these pieces together to present a new algorithm for solving EBUP. We illustrate it on an example in Section 5. Current limitations and avenues for future research are discussed in Section 6.

2. Witnesses for QFBUP

In this section we recall the method for solving QFBUP in [1] which is based on the DLS algorithm [5]:

1. For a quantifier-free input formula φ a disjunctive normal form $C_1 \vee \cdots \vee C_n$ is computed.
2. Then each C_i is of the form $X(\overline{t_1^i}) \wedge \cdots \wedge X(\overline{t_{k_i}^i}) \wedge D_i$ where $X(\overline{t_1^i}), \ldots, X(\overline{t_{k_i}^i})$ are exactly the positive occurrences of X in C_i.
3. Now set $\alpha_i := \lambda \overline{u}.\, \overline{u} \simeq \overline{t_1^i} \vee \cdots \vee \overline{u} \simeq \overline{t_{k_i}^i}$.
4. Return the quantifier-free predicate expression

$$\begin{aligned}
&\lambda \overline{u}.(D_1[X \leftarrow \alpha_1] \rightarrow \alpha_1(\overline{u})) \\
&\quad \wedge (\neg D_1[X \leftarrow \alpha_1] \wedge D_2[X \leftarrow \alpha_2] \rightarrow \alpha_2(\overline{u})) \\
&\quad \wedge \ldots \\
&\quad \wedge (\neg D_1[X \leftarrow \alpha_1] \wedge \cdots \wedge \neg D_{n-1}[X \leftarrow \alpha_{n-1}] \wedge D_n[X \leftarrow \alpha_n] \rightarrow \alpha_n(\overline{u}))
\end{aligned}$$

[1] To find a witness for multiple predicate variables one successively applies the QFBUP-algorithm for each predicate variable.

Denote by γ_φ the output of the above algorithm for input formula φ with respect to the predicate variable X. The central theorem for the correctness of this algorithm is the following:

Theorem 1. *For a quantifier-free formula φ we have $\models_{\text{eq}} \exists X\, \varphi \leftrightarrow \varphi[X \leftarrow \gamma_\varphi]$.*

Proof. See Theorem 5 in [1]. □

We use this result to show a key lemma for our algorithm.

Lemma 2. *Let φ be a quantifier free formula and X a predicate variable. Then $\models_{\text{eq}} \varphi \rightarrow \varphi[X \leftarrow \gamma_\varphi]$.*

Proof. By Theorem 1 we have $\models_{\text{eq}} \exists X\, \varphi \leftrightarrow \varphi[X \leftarrow \gamma_\varphi]$. and in particular $\models_{\text{eq}} \exists X\, \varphi \rightarrow \varphi[X \leftarrow \gamma_\varphi]$. Since X does not occur in $\varphi[X \leftarrow \gamma_\varphi]$ we even get $\models_{\text{eq}} \varphi \rightarrow \varphi[X \leftarrow \gamma_\varphi]$. □

3. Formula Instantiation

Formula instantiation is the following problem: Given a quantifier-free formula φ we want to find a first-order substitution σ such that $\models \varphi\sigma$. In this case we say *σ validates φ*, or *σ is a validating substitution for φ*.

We summarize some results about formula instantiation based on [2] and [6]. Formula instantiation without equality is decidable. More precisely, it is Σ_2^p-complete if the signature contains at least two symbols. The problem is related to several other problems, among them the Herbrand skeleton problem, the matrix instantiation problem, the skeleton instantiation problem and the provability of existential formulae in intuitionistic logic. In first-order logic with equality the problem is undecidable as simultaneous rigid E-unification, which was shown undecidable in [7], can be reduced to it.

For the equality-free case we present a simple, albeit inefficient, method for solving formula instantiation that was also sketched in [6]. The evaluation of more practical algorithms in the literature in the context of EBUP is future work for us.

Given a clause $C = L_1 \vee \cdots \vee L_n$ we define its set of *unification problems* by

$$\mathcal{U}_C := \left\{ (\overline{t}, \overline{s}) \mid \text{ there are } 1 \leq i, j \leq n \text{ and a predicate symbol } P \text{ with } L_i = P(\overline{t}) \text{ and } L_j = \neg P(\overline{s}) \right\}.$$

This means every complementary pair of literals with the same predicate symbol induces a unification problem. Given a finite clause set $N = \{C_1, \ldots, C_m\}$ we define its set of *unification problems* by

$$\mathcal{U}_N := \left\{ \left\{ (\overline{t_1}, \overline{s_1}), \ldots, (\overline{t_m}, \overline{s_m}) \right\} \mid \text{ for all } 1 \leq i \leq m \text{ we have } (\overline{t_i}, \overline{s_i}) \in \mathcal{U}_{C_i} \right\}.$$

A *solution* to a unification problem $\left\{ (\overline{t_1}, \overline{s_1}), \ldots, (\overline{t_m}, \overline{s_m}) \right\} \in \mathcal{U}_N$ is a substitution such σ such that for all $1 \leq i \leq m$ we have $\overline{t_i}\sigma = \overline{s_i}\sigma$. Note that a clause set might contain free first-order variables, but for the purposes of this paper we do not implicitly universally quantify them, but instead let them remain free variables. We say *σ validates N* if σ validates the conjunction of the clauses in N. We now show that σ validates N if and only if σ is the solution for some unification problem in $\mathcal{U}_N$.

Theorem 3. *Let N be a finite clause set and σ a substitution. Then σ validates N if and only if σ is the solution of some $U \in \mathcal{U}_N$.*

Proof. $\Longrightarrow$: Since $\models N\sigma$ we have $\models C\sigma$ for all $C \in N$, i.e., $C\sigma$ contains a complementary pair of literals. Therefore for all $C = L_1 \vee \cdots \vee L_n \in N$ there are terms $\overline{t_C}$, $\overline{s_C}$, $1 \leq i, j \leq n$ and a predicate symbol P such that $L_i = P(\overline{t_C})$ and $L_j = \neg P(\overline{s_C})$ and $\overline{t_C}\sigma = \overline{s_C}\sigma$, i.e. $(\overline{t_C}, \overline{s_C}) \in \mathcal{U}_C$. Therefore σ is a solution for $U := \left\{ (\overline{t_C}, \overline{s_C}) \mid C \in N \right\} \in \mathcal{U}_N$.

$\Longleftarrow$: Let $N = \{C_1, \ldots, C_m\}$ and let σ be a solution for some $U \in \mathcal{U}_N$. Then U is of the form $U = \left\{ (\overline{t_1}, \overline{s_1}), \ldots, (\overline{t_m}, \overline{s_m}) \right\}$ for some terms $\overline{t_1}, \ldots, \overline{t_m}, \overline{s_1}, \ldots, \overline{s_m}$ and for all $1 \leq i \leq m$ we have $(\overline{t_i}, \overline{s_i}) \in \mathcal{U}_{C_i}$ and $\overline{t_i}\sigma = \overline{s_i}\sigma$. Then for all $1 \leq i \leq m$ we have $C_i = L_1 \vee \cdots \vee L_{n_i}$ for some $n_i \in \mathbb{N}$

and since $(\overline{t_i}, \overline{s_i}) \in \mathcal{U}_{C_i}$ there are $1 \leq j, k \leq n_i$ and a predicate symbol P such that $L_j = P(\overline{t_i})$ and $L_k = \neg P(\overline{s_i})$. Now

$$C_i\sigma = L_j\sigma \vee L_k\sigma \vee \bigvee_{\substack{p=1 \\ p\neq j,k}}^{n_i} L_p\sigma = P(\overline{t_i}\sigma) \vee \neg P(\overline{s_i}\sigma) \vee \bigvee_{\substack{p=1 \\ p\neq j,k}}^{n_i} L_p\sigma.$$

Since $\overline{t_i}\sigma = \overline{s_i}\sigma$ we get $\models C_i\sigma$ and therefore $\models N\sigma$. □

Now the algorithm to solve the formula instantiation problem is as follows.

Algorithm 4. *Given a quantifier-free formula $\varphi(\overline{u})$ compute a logically equivalent CNF of $\varphi(\overline{u})$, call it N. Then enumerate all of the finitely many unification problems $U \in \mathcal{U}_N$ and check whether it has a solution σ using a first-order unification algorithm [8]. If such a solution σ is found, then it validates N by Theorem 3 and therefore validates φ. If no solution is found, then by Theorem 3 there is no substitution σ that validates N and thereby none that validates φ.*

4. An Algorithm for EBUP

In this section we give a sound and complete algorithm for solving EBUP. For this, it is useful to consider for $n \in \mathbb{N}$ the following problem which we call EBUP_n: Given a quantifier-free formula $\varphi(\overline{u})$ do there exist terms $\overline{t_1}, \ldots, \overline{t_n}$ and a quantifier-free predicate expression α such that

$$\models_{\mathrm{eq}} \varphi^n(\overline{t_1}, \ldots, \overline{t_n})[X \leftarrow \alpha].$$

In that case we call $\alpha, \overline{t_1}, \ldots, \overline{t_n}$ an EBUP_n*-solution for* $\varphi(\overline{u})$. Since EBUP_n is an instance of the n-Skeleton Herbrand problem [4], it is undecidable, but it serves as a subproblem in our algorithm. As a first step we have the following.

Lemma 5. *If $\alpha, \overline{t_1}, \ldots, \overline{t_n}$ is an EBUP_n-solution for $\varphi(\overline{u})$, then α is an* EBUP*-solution for $\varphi(\overline{u})$.*

Proof. Follows from introducing quantifiers for all terms and rearranging the formula using standard first-order equivalences. □

We will use Herbrand's theorem [9] in the following which holds in first-order logic with and without equality.

Theorem 6 (Herbrand's theorem)**.** *Let $\varphi(\overline{u})$ be a quantifier-free formula. Then $\models_{\mathrm{eq}} \exists \overline{u}\, \varphi(\overline{u})$ ($\models \exists \overline{u}\, \varphi(\overline{u})$) if and only if there exists an $n \in \mathbb{N}$ and terms $\overline{t_1}, \ldots, \overline{t_n}$ such that $\models_{\mathrm{eq}} \varphi^n(\overline{t_1}, \ldots, \overline{t_n})$ ($\models \varphi^n(\overline{t_1}, \ldots, \overline{t_n})$).*

For $\varphi(X, \overline{u})$ quantifier-free and $n \in \mathbb{N}$ denote by $\gamma^n_\varphi(\overline{u_1}, \ldots, \overline{u_n})$ the QFBUP-witness for the formula $\varphi^n(X, \overline{u_1}, \ldots, \overline{u_n})$ obtained by the algorithm from Section 2. Note that the free variables $\overline{u_1}, \ldots, \overline{u_n}$ of φ can occur in the witness γ^n_φ. Using Herbrand's theorem we can show that if an EBUP-solution exists, then there is one that is an instance of $\gamma^n_\varphi(\overline{u_1}, \ldots, \overline{u_n})$ for some $n \in \mathbb{N}$.

Lemma 7. *If $\varphi(\overline{u})$ has an* EBUP*-solution, then there is an $n \in \mathbb{N}$ and $\overline{t_1}, \ldots, \overline{t_n}$ such that $\gamma^n_\varphi(\overline{t_1}, \ldots, \overline{t_n})$, $\overline{t_1}, \ldots, \overline{t_n}$ is an EBUP_n-solution.*

Proof. Assume β is an EBUP-solution for $\varphi(\overline{u})$, i.e.,

$$\models_{\mathrm{eq}} \exists \overline{u}\, \varphi(\overline{u})[X \leftarrow \beta].$$

Then by Herbrand's theorem there exist $n \in \mathbb{N}$ and terms $\overline{t_1}, \ldots, \overline{t_n}$ such that

$$\models_{\mathrm{eq}} (\varphi[X \leftarrow \beta])^n(\overline{t_1}, \ldots, \overline{t_n}). \tag{1}$$

Now the n-fold Herbrand skeleton $\varphi^n(\overline{u_1},\dots,\overline{u_n})$ is a quantifier-free formula in the base language plus the symbols $\overline{u_1},\dots,\overline{u_n}$. By Lemma 2 we get

$$\models_{\mathrm{eq}} \varphi^n(\overline{u_1},\dots,\overline{u_n}) \to \varphi^n(\overline{u_1},\dots,\overline{u_n})[X \leftarrow \gamma^n_\varphi(\overline{u_1},\dots,\overline{u_n})].$$

By applying the substitution $[X \leftarrow \beta, \overline{u_1} \leftarrow \overline{t_1},\dots,\overline{u_n} \leftarrow \overline{t_n}]$ we have

$$\models_{\mathrm{eq}} \varphi^n(\overline{t_1},\dots,\overline{t_n})[X \leftarrow \beta] \to \varphi^n(\overline{t_1},\dots,\overline{t_n})[X \leftarrow \gamma^n_\varphi(\overline{t_1},\dots,\overline{t_n})]$$

Note that $\varphi^n(\overline{t_1},\dots,\overline{t_n})[X \leftarrow \beta] = (\varphi[X \leftarrow \beta])^n(\overline{t_1},\dots,\overline{t_n})$ so together with (1) we get

$$\models_{\mathrm{eq}} \varphi^n(\overline{t_1},\dots,\overline{t_n})[X \leftarrow \gamma^n_\varphi(\overline{t_1},\dots,\overline{t_n})],$$

i.e., $\gamma^n_\varphi(\overline{t_1},\dots,\overline{t_n}), \overline{t_1},\dots,\overline{t_n}$ is an EBUP_n-solution for φ. □

To handle equality reasoning in first-order logic without equality let T_{eq} be a finite universal first-order axiomatization of equality, i.e., we have $T_{\mathrm{eq}} = \forall \overline{v}\, \psi_{\mathrm{eq}}(\overline{v})$ for some quantifier-free $\psi_{\mathrm{eq}}(\overline{v})$. For example, in a language with function symbols $f_1,\dots,f_n$ and predicate symbols $P_1,\dots,P_m$ we have

$$\begin{aligned}
\psi_{\mathrm{eq}}(\overline{v}) = {} & v^{\mathrm{refl}} \simeq v^{\mathrm{refl}} \\
& \wedge (v_1^{\mathrm{sym}} \simeq v_2^{\mathrm{sym}} \to v_2^{\mathrm{sym}} \simeq v_1^{\mathrm{sym}}) \\
& \wedge (v_1^{\mathrm{trans}} \simeq v_2^{\mathrm{trans}} \wedge v_2^{\mathrm{trans}} \simeq v_3^{\mathrm{trans}} \to v_1^{\mathrm{trans}} \simeq v_3^{\mathrm{trans}}) \\
& \wedge \bigwedge_{i=1}^{n} (v_1^{f_i} \simeq v_2^{f_i} \to f(v_1^{f_i}) \simeq f(v_2^{f_i})) \\
& \wedge \bigwedge_{i=1}^{m} (v_1^{P_i} \simeq v_2^{P_i} \to (P(v_1^{P_i}) \leftrightarrow P(v_2^{P_i}))).
\end{aligned}$$

We then have $\models_{\mathrm{eq}} \varphi$ if and only if $\models \forall \overline{v}\, \psi_{\mathrm{eq}}(\overline{v}) \to \varphi$.

Using Herbrand's theorem again we show that if there is an EBUP-solution for $\exists \overline{u}\, \varphi(\overline{u})$ then $(*)$ is valid in first-order logic without equality for some $n, k \in \mathbb{N}$ and terms $\overline{t_1},\dots,\overline{t_n}, \overline{s_1},\dots,\overline{s_k}$.

Lemma 8. *If $\varphi(\overline{u})$ has an* EBUP*-solution, then there exist $n, k \in \mathbb{N}$ and $\overline{t_1},\dots,\overline{t_n}, \overline{s_1},\dots,\overline{s_k}$ such that*

$$\models \psi_{\mathrm{eq}}(\overline{s_1}) \wedge \dots \wedge \psi_{\mathrm{eq}}(\overline{s_k}) \to \varphi^n(\overline{t_1},\dots,\overline{t_n})[X \leftarrow \gamma^n_\varphi(\overline{t_1},\dots,\overline{t_n})].$$

Proof. By Lemma 7 we get

$$\models_{\mathrm{eq}} \varphi^n(\overline{t_1},\dots,\overline{t_n})[X \leftarrow \gamma^n_\varphi(\overline{t_1},\dots,\overline{t_n})]$$

for some $n \in \mathbb{N}$ and some terms $\overline{t_1},\dots,\overline{t_n}$. We then get

$$\models \forall \overline{v}\, \psi_{\mathrm{eq}}(\overline{v}) \to \varphi^n(\overline{t_1},\dots,\overline{t_n})[X \leftarrow \gamma^n_\varphi(\overline{t_1},\dots,\overline{t_n})]$$

which is equivalent to

$$\models \exists \overline{v}\, (\psi_{\mathrm{eq}}(\overline{v}) \to \varphi^n(\overline{t_1},\dots,\overline{t_n})[X \leftarrow \gamma^n_\varphi(\overline{t_1},\dots,\overline{t_n})]).$$

By Herbrand's theorem and standard propositional equivalences there exists a $k \in \mathbb{N}$ and terms $\overline{s_1},\dots,\overline{s_k}$ such that

$$\models \psi_{\mathrm{eq}}(\overline{s_1}) \wedge \dots \wedge \psi_{\mathrm{eq}}(\overline{s_k}) \to \varphi^n(\overline{t_1},\dots,\overline{t_n})[X \leftarrow \gamma^n_\varphi(\overline{t_1},\dots,\overline{t_n})].$$

□

This now leads us to the description of our algorithm.

Algorithm 9. *Enumerate all* $(n,k) \in \mathbb{N}^2$ *and for each* (n,k) *use Algorithm 4 to find terms* $\overline{t_1},\dots,\overline{t_n}$, $\overline{s_1},\dots,\overline{s_k}$ *such that*

$$\models \bigwedge_{i=1}^{k} \psi_{\mathrm{eq}}(\overline{s_i}) \to (\varphi[X \leftarrow \gamma^n_\varphi(\overline{u_1},\dots,\overline{u_n})])^n(\overline{t_1},\dots,\overline{t_n}),$$

if they exist. If yes, return $\gamma^n_\varphi(\overline{t_1},\dots,\overline{t_n})$*. Otherwise, continue with the next pair* (n,k)*.*

Algorithm 9 is sound for EBUP.

Lemma 10. *If Algorithm 9 terminates on input* $\varphi(\overline{u})$*, then its result is an* EBUP*-solution for* $\varphi(\overline{u})$*.*

Proof. If Algorithm 9 terminates, then there are $n,k \in \mathbb{N}$ and terms $\overline{t_1},\dots,\overline{t_n}$, $\overline{s_1},\dots,\overline{s_k}$ such that

$$\models \bigwedge_{i=1}^{k} \psi_{\mathrm{eq}}(\overline{s_i}) \to (\varphi[X \leftarrow \gamma^n_\varphi(\overline{u_1},\dots,\overline{u_n})])^n(\overline{t_1},\dots,\overline{t_n})$$

and the result of Algorithm 9 is $\gamma^n_\varphi(\overline{t_1},\dots,\overline{t_n})$. By introducing the quantifiers in the equality theory we get

$$\models T_{\mathrm{eq}} \to (\varphi[X \leftarrow \gamma^n_\varphi(\overline{u_1},\dots,\overline{u_n})])^n(\overline{t_1},\dots,\overline{t_n})$$

or, in other words,

$$\models_{\mathrm{eq}} (\varphi[X \leftarrow \gamma^n_\varphi(\overline{u_1},\dots,\overline{u_n})])^n(\overline{t_1},\dots,\overline{t_n}).$$

Note that

$$\begin{aligned}(\varphi[X \leftarrow \gamma^n_\varphi(\overline{u_1},\dots,\overline{u_n})])^n(\overline{t_1},\dots,\overline{t_n}) &= (\varphi[X \leftarrow \gamma^n_\varphi(\overline{t_1},\dots,\overline{t_n})])^n(\overline{t_1},\dots,\overline{t_n}) \\ &= \varphi^n(\overline{t_1},\dots,\overline{t_n})[X \leftarrow \gamma^n_\varphi(\overline{t_1},\dots,\overline{t_n})].\end{aligned}$$

Therefore $\gamma^n_\varphi(\overline{t_1},\dots,\overline{t_n}),\overline{t_1},\dots,\overline{t_n}$ is an EBUP_n-solution for $\varphi(\overline{u})$ and thus an EBUP-solution for $\varphi(\overline{u})$ by Lemma 5. □

Algorithm 9 is also complete for EBUP.

Lemma 11. *Algorithm 9 is complete for* EBUP*, i.e., if* $\varphi(\overline{u})$ *has an* EBUP*-solution, then Algorithm 9 terminates.*

Proof. If $\varphi(\overline{u})$ has an EBUP-solution then by Lemma 8 there are $n,k \in \mathbb{N}$ and terms $\overline{t_1},\dots,\overline{t_n}$, $\overline{s_1},\dots,\overline{s_k}$ such that

$$\models \bigwedge_{i=1}^{k} \psi_{\mathrm{eq}}(\overline{s_i}) \to \varphi^n(\overline{t_1},\dots,\overline{t_n})[X \leftarrow \gamma^n_\varphi(\overline{t_1},\dots,\overline{t_n})].$$

Again, note that

$$\varphi^n(\overline{t_1},\dots,\overline{t_n})[X \leftarrow \gamma^n_\varphi(\overline{t_1},\dots,\overline{t_n})] = (\varphi[X \leftarrow \gamma^n_\varphi(\overline{u_1},\dots,\overline{u_n})])^n(\overline{t_1},\dots,\overline{t_n}).$$

Thus

$$\models \bigwedge_{i=1}^{k} \psi_{\mathrm{eq}}(\overline{s_i}) \to (\varphi[X \leftarrow \gamma^n_\varphi(\overline{u_1},\dots,\overline{u_n})])^n(\overline{t_1},\dots,\overline{t_n})$$

which means that Algorithm 9 terminates. □

It follows that Algorithm 9 is a sound and complete algorithm for EBUP.

Theorem 12. *Let* $\varphi(\overline{u})$ *be quantifier-free. Then Algorithm 9 terminates with an* EBUP*-solution for* $\varphi(\overline{u})$ *if and only if* $\varphi(\overline{u})$ *has an* EBUP*-solution.*

Proof. Follows from Lemmas 10 and 11. □

5. Example

We illustrate the algorithm on a simple example. Consider the formula

$$\varphi(u_1, u_2) = 0 \not\simeq s(u_1) \to X(u_2) \wedge \neg X(s(u_2)).$$

We are interested in finding a witness α for $\exists u_1\, \exists u_2\, \varphi(u_1, u_2)$. We apply Algorithm 9 to this problem. First consider $n = 0$. Then $\varphi^n \equiv \bot$ so this case is not interesting.

Now consider $n = 1$ and $k = 0$. We first compute γ^1_φ. To that end we need a DNF φ' of φ^1 which is given by

$$0 \simeq s(u_1) \vee X(u_2) \wedge \neg X(s(u_2)).$$

Since the only disjunct containing X is the last one we apply the QFBUP construction only to it and get $\gamma^1_\varphi = \lambda w.s(u_2) \not\simeq u_2 \to w \simeq u_2$. Next we solve the formula instantiation problem for $\bigwedge_{i=1}^k \psi_{\mathrm{eq}}(\overline{v_i}) \to \varphi[X \leftarrow \gamma^1_\varphi]$ using the algorithm from Section 3. Since $k = 0$ the conjunction $\bigwedge_{i=1}^k \psi_{\mathrm{eq}}(\overline{v_i})$ vanishes. The quantifier-free formula $\varphi[X \leftarrow \gamma^1_\varphi]$ is given by

$$0 \not\simeq s(u_1) \to (s(u_2) \not\simeq u_2 \to u_2 \simeq u_2) \wedge \neg(s(u_2) \not\simeq u_2 \to s(u_2) \simeq u_2).$$

A CNF of this formula is given by

$$(0 \simeq s(u_1) \vee s(u_2) \simeq u_2 \vee u_2 \simeq u_2) \wedge (0 \simeq s(u_1) \vee s(u_2) \not\simeq u_2).$$

This does not have a solution as the first clause only contains positive literals. Thus for $n = 1, k = 0$ there is no solution. We now check $n = 1, k = 1$. Now $\bigwedge_{i=1}^k \psi_{\mathrm{eq}}(\overline{v_i}) \to \varphi[X \leftarrow \gamma^1_\varphi]$ is given by

$$\psi_{\mathrm{eq}}(\overline{v}) \to (0 \not\simeq s(u_1) \to (s(u_2) \not\simeq u_2 \to u_2 \simeq u_2) \wedge \neg(s(u_2) \not\simeq u_2 \to s(u_2) \simeq u_2)).$$

Computing a CNF of this formula produces a lot of clauses, in part due to the equality axioms. However, in this case we will only need reflexivity and symmetry to show the existence of a validating substitution so for the purposes of this example we only use those two axioms. A validating substitution for this smaller equality theory is still a validating substitution for the full equality theory. Thus we consider the formula

$$\begin{aligned}&(v^{\mathrm{refl}} \simeq v^{\mathrm{refl}} \wedge (v_1^{\mathrm{sym}} \simeq v_2^{\mathrm{sym}} \to v_2^{\mathrm{sym}} \simeq v_1^{\mathrm{sym}}))\\ &\quad \to (0 \not\simeq s(u_1) \to (s(u_2) \not\simeq u_2 \to u_2 \simeq u_2) \wedge \neg(s(u_2) \not\simeq u_2 \to s(u_2) \simeq u_2)).\end{aligned}$$

A CNF of this formula is given by the clause set N with the clauses

$$\begin{aligned}C_1 &= v^{\mathrm{refl}} \not\simeq v^{\mathrm{refl}} \vee v_1^{\mathrm{sym}} \simeq v_2^{\mathrm{sym}} \vee 0 \simeq s(u_1) \vee s(u_2) \simeq u_2 \vee u_2 \simeq u_2,\\ C_2 &= v^{\mathrm{refl}} \not\simeq v^{\mathrm{refl}} \vee v_2^{\mathrm{sym}} \not\simeq v_1^{\mathrm{sym}} \vee 0 \simeq s(u_1) \vee s(u_2) \simeq u_2 \vee u_2 \simeq u_2,\\ C_3 &= v^{\mathrm{refl}} \not\simeq v^{\mathrm{refl}} \vee v_1^{\mathrm{sym}} \simeq v_2^{\mathrm{sym}} \vee 0 \simeq s(u_1) \vee s(u_2) \not\simeq u_2,\\ C_4 &= v^{\mathrm{refl}} \not\simeq v^{\mathrm{refl}} \vee v_2^{\mathrm{sym}} \not\simeq v_1^{\mathrm{sym}} \vee 0 \simeq s(u_1) \vee s(u_2) \not\simeq u_2.\end{aligned}$$

Now one has to check all the unification problems in $\mathcal{U}_N$ to find a σ that validates N. We give one unification problem in $\mathcal{U}_N$ that has a solution: Note that $U_1 = ((v^{\mathrm{refl}}, v^{\mathrm{refl}}), (u_2, u_2)) \in \mathcal{U}_{C_1} \cap \mathcal{U}_{C_2}$ as well as $U_2 = ((v_1^{\mathrm{sym}}, v_2^{\mathrm{sym}}), (s(u_2), u_2)) \in \mathcal{U}_{C_3}$ and $U_3 = ((0, s(u_1)), (v_2^{\mathrm{sym}}, v_1^{\mathrm{sym}})) \in \mathcal{U}_{C_4}$. Then a solvable problem in $\mathcal{U}_N$ is given by $U = \{U_1, U_2, U_3\}$ which corresponds to the unification problem

$$v^{\mathrm{refl}} = u_2 \qquad v_1^{\mathrm{sym}} = s(u_2) \qquad v_2^{\mathrm{sym}} = u_2 \qquad 0 = v_2^{\mathrm{sym}} \qquad s(u_1) = v_1^{\mathrm{sym}}.$$

This has a solution given by

$$\sigma = [v^{\mathrm{refl}} \leftarrow 0, v_1^{\mathrm{sym}} \leftarrow s(0), v_2^{\mathrm{sym}} \leftarrow 0, u_1 \leftarrow 0, u_2 \leftarrow 0].$$

Applying this substitution to γ^1_φ results in the witness $\lambda w.s(0) \not\simeq 0 \to w \simeq 0$.

6. Discussion and Conclusion

We presented a first version of a practical sound and complete algorithm for solving existential Boolean unification with predicates. The algorithm combines the QFBUP-witness construction and formula instantiation to produce quantifier-free witnesses. To our knowledge this is the first algorithm for EBUP that goes beyond the naive enumeration of first-order predicates.

There is a lot of room for improvement and avenues for further research, in particular concerning the efficiency of the algorithm. There are currently three stages in the algorithm that incur an exponential blowup on their own. The first is the computation of the QFBUP-witness whose size is exponential in the size of the input formula [1]. The second one is the computation of the CNF for solving the formula instantiation problem. The third one is the number of unification problems that need to be checked, given a CNF. Thus, avenues for improving the efficiency include looking into finding smaller witnesses for QFBUP and evaluating existing algorithms for formula instantiation in the literature for the use in the context of EBUP.

Further improvements to the algorithm might include giving more fine-grained control over the amount of instances to allow per existential quantifier. Currently we only have two parameters: n controls the number of instances of the quantifiers in the input formula, k controls the number of instances of the whole first-order equality theory. In our example we already used a simplification where we only instantiated part of the first-order equality theory to improve the search for validating substitutions. One could take this idea further and introduce a separate parameter for every quantifier or axiom so as to independently control how many instances of a given axiom are allowed to find a Herbrand expansion. This is especially useful if one expects only a small amount of instances of these axioms to be necessary for showing the validity of the input formula. Providing these parameters also allows a more fine-grained ramp-up in computational resources needed while the algorithm iterates.

Declaration on Generative AI

The authors have not employed any Generative AI tools.

References

[1] S. Eberhard, S. Hetzl, D. Weller, Boolean unification with predicates, Journal of Logic and Computation 27 (2017) 109–128. doi:10.1093/logcom/exv059.

[2] A. Voronkov, Simultaneous rigid e-unification and other decision problems related to the herbrand theorem, Theoretical Computer Science 224 (1999) 319–352. doi:10.1016/S0304-3975(98)00317-X.

[3] A. Degtyarev, A. Voronkov, The undecidability of simultaneous rigid e-unification, Theoretical Computer Science 166 (1996) 291–300. doi:10.1016/0304-3975(96)00092-8.

[4] Y. Gurevich, M. Veanes, Logic with equality: Partisan corroboration and shifted pairing, Information and Computation 152 (1999) 205–235. doi:10.1006/inco.1999.2797.

[5] P. Doherty, W. Lukaszewicz, A. Szalas, Computing circumscription revisited: A reduction algorithm, Journal of Automated Reasoning 18 (1997) 297–336. doi:10.1023/A:1005722130532.

[6] A. Degtyarev, Y. Gurevich, A. Voronkov, Herbrand's theorem and equational reasoning: Problems and solutions, in: G. Paun, G. Rozenberg, A. Salomaa (Eds.), Current Trends in Theoretical Computer Science, Entering the 21th Century, World Scientific, 2001, pp. 303–326.

[7] A. Degtyarev, A. Voronkov, Simultaneous rigid e-unification is undecidable, in: H. Kleine Büning (Ed.), Computer Science Logic, Springer Berlin Heidelberg, Berlin, Heidelberg, 1996, pp. 178–190. doi:10.1007/3-540-61377-3_38.

[8] J. A. Robinson, A machine-oriented logic based on the resolution principle, J. ACM 12 (1965) 23–41. doi:10.1145/321250.321253.

[9] J. Herbrand, Logical Writings, Harvard University Press, Cambridge, MA, 1971.