Proof search in minimal logic

Helmut Schwichtenberg

Mathematisches Institut der Universität München,
schwicht@mathematik.uni-muenchen.de,
WWW home page: http://www.mathematik.uni-muenchen.de/~schwicht/

1 Introduction

We describe a rather natural proof search algorithm for a certain fragment of higher order (simply typed) minimal logic. This fragment is determined by requiring that every higher order variable Y can only occur in a context $Y\boldsymbol{x}$, where $\boldsymbol{x}$ are distinct bound variables in the scope of the operator binding Y, and of opposite polarity. Note that for first order logic this restriction does not mean anything, since there are no higher order variables. However, when designing a proof search algorithm for first order logic only, one is naturally led into this fragment of higher order logic, where the algorithm works as well.

In doing this we rely heavily on Miller's [1], who has introduced this type of restriction to higher order terms (called *patterns* by Nipkow [2]), noted its relevance for extensions of logic programming, and showed that the unification problem for patterns is solvable and admits most general unifiers. The present paper was motivated by the desire to use Miller's approach as a basis for an implementation of a simple proof search engine for (first and higher order) minimal logic. This goal prompted us into several simplifications, optimizations and extensions, in particular the following.

- Instead of arbitrarily mixed prefixes we only use those of the form $\forall\exists\forall$. Nipkow in [2] already had presented a version of Miller's pattern unification algorithm for such prefixes, and Miller in [1, Section 9.2] notes that in such a situation any two unifiers can be transformed into each other by a variable renaming substitution. Here we restrict ourselves to $\forall\exists\forall$-prefixes throughout, i.e., in the proof search algorithm as well.
- The order of events in the pattern unification algorithm is changed slightly, by postponing the raising step until it is really needed. This avoids unnecessary creation of new higher type variables. – Already Miller noted in [1, p.515] that such optimizations are possible.
- The extensions concern the (strong) existential quantifier, which has been left out in Miller's treatment, and also conjunction. The latter can be avoided in principle, but of course is a useful thing to have.

Moreover, since part of the motivation to write this paper was the necessity to have a guide for our implementation, we have paid particular attention to write at least the parts of the proofs with algorithmic content as clear and complete as possible.

The paper is organized as follows. Section 2 defines the pattern unification algorithm, and in section 3 its correctness and completeness is proved. Section 4 presents the proof search algorithm, and again its correctness and completeness is proved. The final section 5 contains what we have to say about extensions to $\wedge$ and $\exists$.

2 The unification algorithm

We work in the simply typed λ-calculus, with the usual conventions. For instance, whenever we write a term we assume that it is correctly typed. *Substitutions* are denoted by φ, ψ, ρ. The result of applying a substitution φ to a term t or a formula A is written as $t\varphi$ or $A\varphi$, with the understanding that after the substitution all terms are brought into long normal form.

Q always denotes a $\forall\exists\forall$-prefix, say $\forall \boldsymbol{x} \exists \boldsymbol{y} \forall \boldsymbol{z}$, with distinct variables. We call $\boldsymbol{x}$ the *signature variables*, $\boldsymbol{y}$ the *flexible variables* and $\boldsymbol{z}$ the *forbidden variables* of Q, and write $Q_\exists$ for the existential part $\exists \boldsymbol{y}$ of Q.

Q-terms are inductively defined by the following clauses.

- If u is a universally quantified variable in Q or a constant, and $\boldsymbol{r}$ are Q-terms, then $u\boldsymbol{r}$ is a Q-term.
- For any flexible variable y and distinct forbidden variables $\boldsymbol{z}$ from Q, $y\boldsymbol{z}$ is a Q-term.
- If r is a $Q\forall z$-term, then $\lambda z r$ is a Q-term.

Explicitely, r is a Q-term iff all its free variables are in Q, and for every subterm $y\boldsymbol{r}$ of r with y free in r and flexible in Q, the $\boldsymbol{r}$ are distinct variables either λ-bound in r (such that $y\boldsymbol{r}$ is in the scope of this λ) or else forbidden in Q.

Q-goals and *Q-clauses* are simultaneously defined by

- If $\boldsymbol{r}$ are Q-terms, then $P\boldsymbol{r}$ is a Q-goal as well as a Q-clause.
- If D is a Q-clause and G is a Q-goal, then $D \to G$ is a Q-goal.
- If G is a Q-goal and D is a Q-clause, then $G \to D$ is a Q-clause.
- If G is a $Q\forall x$-goal, then $\forall x G$ is a Q-goal.
- If $D[y := Y\boldsymbol{z}]$ is a $\forall \boldsymbol{x} \exists \boldsymbol{y}, Y \forall \boldsymbol{z}$-clause, then $\forall y D$ is a $\forall \boldsymbol{x} \exists \boldsymbol{y} \forall \boldsymbol{z}$-clause.

Explicitely, a formula A is a *Q-goal* iff all its free variables are in Q, and for every subterm $y\boldsymbol{r}$ of A with y either existentially bound in A (with $y\boldsymbol{r}$ in the scope) or else free in A and flexible in Q, the $\boldsymbol{r}$ are distinct variables either λ- or universally bound in A (such that $y\boldsymbol{r}$ is in the scope) or else free in A and forbidden in Q.

A *Q-substitution* is a substitution of Q-terms.

A *unification problem $\mathcal{U}$* consists of a $\forall\exists\forall$-prefix Q and a conjunction C of equations between Q-terms of the same type, i.e., $\bigwedge_{i=1}^{n} r_i = s_i$. We may assume that each such equation is of the form $\lambda \boldsymbol{x} r = \lambda \boldsymbol{x} s$ with the same $\boldsymbol{x}$ (which may be empty) and r, s of ground type.

A *solution* to such a unification problem $\mathcal{U}$ is a Q-substitution φ such that for every i, $r_i\varphi = s_i\varphi$ holds (i.e., $r_i\varphi$ and $s_i\varphi$ have the same normal form). We sometimes write C as $\boldsymbol{r} = \boldsymbol{s}$, and (for obvious reasons) call it a list of unification pairs.

We now define the unification algorithm. It takes a unification problem $\mathcal{U} = QC$ and returns a substitution ρ and another patter unification problem $\mathcal{U}' = Q'C'$. Note that ρ will be neither a Q-substitution nor a Q'-substitution, but will have the property that

- ρ is defined on flexible variables of Q only, and its value terms have no free occurrences of forbidden variables from Q,
- if G is a Q-goal, then $G\rho$ is a Q'-goal, and
- whenever φ' is an $\mathcal{U}'$-solution, then $(\rho \circ \varphi')\restriction Q_\exists$ is an $\mathcal{U}$-solution.

To define the unification algorithm, we distinguish cases according to the form of the unification problem, and either give the transition done by the algorithm, or else state that it fails.

Case identity, i.e., $Q.r = r \wedge C$. Then

$$Q.r = r \wedge C \Longrightarrow_\varepsilon QC.$$

Case ξ, i.e., $Q.\lambda\boldsymbol{x}\, r = \lambda\boldsymbol{x}\, s \wedge C$. We may assume here that the bound variables $\boldsymbol{x}$ are the same on both sides.

$$Q.\lambda\boldsymbol{x}\, r = \lambda\boldsymbol{x}\, s \wedge C \Longrightarrow_\varepsilon Q\forall\boldsymbol{x}.r = s \wedge C.$$

Case rigid-rigid, i.e., $Q.f\boldsymbol{r} = f\boldsymbol{s} \wedge C$.

$$Q.f\boldsymbol{r} = f\boldsymbol{s} \wedge C \Longrightarrow_\varepsilon Q.\boldsymbol{r} = \boldsymbol{s} \wedge C.$$

Case flex-flex with equal heads, i.e., $Q.u\boldsymbol{y} = u\boldsymbol{z} \wedge C$.

$$Q.u\boldsymbol{y} = u\boldsymbol{z} \wedge C \Longrightarrow_\rho Q'.C\rho$$

with $\rho = [u := \lambda\boldsymbol{y}.u'\boldsymbol{w}]$, Q' is Q with $\exists u$ replaced by $\exists u'$, and $\boldsymbol{w}$ an enumeration of $\{\, y_i \mid y_i = z_i \,\}$ (note $\lambda\boldsymbol{y}.u'\boldsymbol{w} = \lambda\boldsymbol{z}.u'\boldsymbol{w}$).

Case flex-flex with different heads, i.e., $Q.u\boldsymbol{y} = v\boldsymbol{z} \wedge C$.

$$Q.u\boldsymbol{y} = v\boldsymbol{z} \wedge C \Longrightarrow_\rho Q'C\rho,$$

where ρ and Q' are defined as follows. Let $\boldsymbol{w}$ be an enumeration of the variables both in $\boldsymbol{y}$ and in $\boldsymbol{z}$. Then $\rho = [u, v := \lambda\boldsymbol{y}.u'\boldsymbol{w}, \lambda\boldsymbol{z}.u'\boldsymbol{w}]$, and Q' is Q with $\exists u, \exists v$ removed and $\exists u'$ inserted.

Case flex-rigid, i.e., $Q.u\boldsymbol{y} = t \wedge C$ with t rigid, i.e., not of the form $v\boldsymbol{z}$ with flexible v.

Subcase occurrence check: t contains (a critical subterm with head) u. Fail.

Subcase pruning: t contains a subterm $v\boldsymbol{w}_1 z\boldsymbol{w}_2$ with $\exists v$ in Q, and z free in t but not in $\boldsymbol{y}$.

$$Q.u\boldsymbol{y} = t \wedge C \Longrightarrow_\rho Q'.u\boldsymbol{y} = t\rho \wedge C\rho$$

where $\rho = [v := \lambda\boldsymbol{w}_1, z, \boldsymbol{w}_2.v'\boldsymbol{w}_1\boldsymbol{w}_2]$, Q' is Q with $\exists v$ replaced by $\exists v'$.

Subcase pruning impossible: $\lambda \boldsymbol{y} t$ (after all pruning steps are done still) has a free occurrence of a forbidden variable z. Fail.

Subcase explicit definition: otherwise.

$$Q.u\boldsymbol{y} = t \wedge C \Longrightarrow_\rho Q'C\rho$$

where $\rho = [u := \lambda \boldsymbol{y} t]$, and Q' is obtained from Q by removing $\exists u$.

This concludes the definition of the unification algorithm.

Our next task is to prove that this algorithm indeed has the three properties stated above. The first one (ρ is defined on flexible variables of Q only, and its value terms have no free occurrences of forbidden variables from Q) is obvious from the definition. We now prove the second one; the third one will be proved in the next section.

Lemma 1. *If $Q \Longrightarrow_\rho Q'$ and G is a Q-goal, then $G\rho$ is a Q'-goal.*

Proof. We distinguish cases according to the definition of the unification algorithm.

Cases identity, ξ and rigid-rigid. Then $\rho = \varepsilon$ and the claim is trivial.

Case flex-flex with equal heads. Then $\rho = [u := \lambda \boldsymbol{y}.u'\boldsymbol{w}]$ with $\boldsymbol{w}$ a sublist of $\boldsymbol{y}$, and Q' is Q with $\exists u$ replaced by $\exists u'$. Then clearly $G[u := \lambda \boldsymbol{y}.u'\boldsymbol{w}]$ is a Q'-goal (recall that after a substitution we always normalize).

Case flex-flex with different heads. Then $\rho = [u, v := \lambda \boldsymbol{y}.u'\boldsymbol{w}, \lambda \boldsymbol{z}.u'\boldsymbol{w}]$ with $\boldsymbol{w}$ an enumeration of the variables both in $\boldsymbol{y}$ and in $\boldsymbol{z}$, and Q' is Q with $\exists u, \exists v$ removed and $\exists u'$ inserted. Again clearly $G[u, v := \lambda \boldsymbol{y}.u'\boldsymbol{w}, \lambda \boldsymbol{z}.u'\boldsymbol{w}]$ is a Q'-goal.

Case flex-rigid, *Subcase* pruning: Then $\rho = [v := \lambda \boldsymbol{w}_1, z, \boldsymbol{w}_2.v'\boldsymbol{w}_1\boldsymbol{w}_2]$, and Q' is Q with $\exists v$ replaced by $\exists v'$. Suppose G is a Q-goal. Then clearly $G[v := \lambda \boldsymbol{w}_1, z, \boldsymbol{w}_2.v'\boldsymbol{w}_1\boldsymbol{w}_2]$ is a Q'-goal.

Case flex-rigid, *Subcase* explicit definition: Then $\rho = [u := \lambda \boldsymbol{y} t]$ with a Q-term $\lambda \boldsymbol{y} t$ without free occurrences of forbidden variables, and Q' is obtained from Q by removing $\exists u$. Suppose G is a Q-goal. Then clearly $G[u := \lambda \boldsymbol{y} t]$ form) is a Q'-goal.

Let $Q \longrightarrow_\rho Q'$ mean that for some C, C' we have $QC \Longrightarrow_\rho Q'C'$. Write $Q \longrightarrow^*_\rho Q'$ if there are $\rho_1, \dots, \rho_n$ and $Q_1, \dots, Q_{n-1}$ such that

$$Q \longrightarrow_{\rho_1} Q_1 \longrightarrow_{\rho_2} \dots \longrightarrow_{\rho_{n-1}} Q_{n-1} \longrightarrow_{\rho_n} Q',$$

and $\rho = \rho_1 \circ \dots \circ \rho_n$.

Corollary 1. *If $Q \longrightarrow^*_\rho Q'$ and G is a Q-goal, then $G\rho$ is a Q'-goal.*

3 Correctness and completeness of the unification algorithm

Lemma 2. *Let a unification problem $\mathcal{U}$ consisting of a $\forall\exists\forall$-prefix Q and a list $\boldsymbol{r} = \boldsymbol{s}$ of unification pairs be given. Then either*

– *the unification algorithm makes a transition $\mathcal{U} \Longrightarrow_\rho \mathcal{U}'$, and*

$$\begin{aligned} \Phi' : \mathcal{U}'\text{-}solutions &\to \mathcal{U}\text{-}solutions \\ \varphi' &\mapsto (\rho \circ \varphi')\restriction Q_\exists \end{aligned}$$

is well-defined and we have $\Phi\colon \mathcal{U}$-solutions $\to \mathcal{U}'$-solutions such that Φ' is inverse to Φ, i.e. $\Phi'(\Phi\varphi) = \varphi$, or else

– *the unification algorithm fails, and there is no $\mathcal{U}$-solution.*

Proof. Case identity, i.e., $Q.r = r \wedge C \Longrightarrow_\varepsilon QC$. Let Φ be the identity.

Case ξ, i.e., $Q.\lambda\boldsymbol{x}\,r = \lambda\boldsymbol{x}\,s \wedge C \Longrightarrow_\varepsilon Q\forall\boldsymbol{x}.r = s \wedge C$. Let again Φ be the identity.

Case rigid-rigid, i.e., $Q.f\boldsymbol{r} = f\boldsymbol{s} \wedge C \Longrightarrow_\varepsilon Q.\boldsymbol{r} = \boldsymbol{s} \wedge C$. Let again Φ be the identity.

Case flex-flex with equal heads, i.e., $Q.u\boldsymbol{y} = u\boldsymbol{z} \wedge C \Longrightarrow_\rho Q'.C\rho$ with $\rho = [u := \lambda\boldsymbol{y}.u'\boldsymbol{w}]$, Q' is Q with $\exists u$ replaced by $\exists u'$, and $\boldsymbol{w}$ an enumeration of those y_i which are identical to z_i (i.e., the variable at the same position in $\boldsymbol{z}$). Notice that $\lambda\boldsymbol{y}.u'\boldsymbol{w} = \lambda\boldsymbol{z}.u'\boldsymbol{w}$.

1. Φ' is well-defined: Let φ' be a $\mathcal{U}'$-solution, i.e., assume that $C\rho\varphi'$ holds. We must show that $\varphi := (\rho \circ \varphi')\restriction Q_\exists$ is a $\mathcal{U}$-solution.

For $u\boldsymbol{y} = u\boldsymbol{z}$: We must show $(u\varphi)\boldsymbol{y} = (u\varphi)\boldsymbol{z}$. But $u\varphi = u\rho\varphi' = (\lambda\boldsymbol{y}.u'\boldsymbol{w})\varphi'$. Hence $(u\varphi)\boldsymbol{y} = (u\varphi)\boldsymbol{z}$ by the construction of $\boldsymbol{w}$.

For $(r = s) \in C$: We need to show $(r = s)\varphi$. But by assumption $(r = s)\rho\varphi'$ holds, and $r = s$ has all its flexible variables from $Q_\exists$.

2. Definition of $\Phi\colon \mathcal{U}$-solutions $\to \mathcal{U}'$-solutions. Let a Q-substitution φ be given such that $(u\boldsymbol{y} = u\boldsymbol{z})\varphi$ and $C\varphi$. Define $u'(\Phi\varphi) := \lambda\boldsymbol{w}.(u\varphi)\boldsymbol{w0}$ (w.l.o.g), and $v(\Phi\varphi) := v$ for every other variable v in $Q_\exists$.

$\Phi\varphi =: \varphi'$ is a $\mathcal{U}'$-solution: Let $(r = s) \in C$. Then $(r = s)\varphi$ by assumption, for φ is a Q-substitution such that $C\varphi$ holds. We must show

$$(r = s)\rho\varphi'.$$

Notice that our assumption $(u\varphi)\boldsymbol{y} = (u\varphi)\boldsymbol{z}$ implies that the normal form of both sides can only contain the variables in $\boldsymbol{w}$. Therefore

$$\begin{aligned} u\rho\varphi' &= (\lambda\boldsymbol{y}.u'\boldsymbol{w})\varphi' \\ &= \lambda\boldsymbol{y}.(\lambda\boldsymbol{w}.(u\varphi)\boldsymbol{w0})\boldsymbol{w} \\ &= \lambda\boldsymbol{y}.(u\varphi)\boldsymbol{w0} \\ &= \lambda\boldsymbol{y}.(u\varphi)\boldsymbol{y} \\ &= u\varphi \end{aligned}$$

and hence $(r = s)\rho\varphi'$.

3. $\Phi'(\Phi\varphi) = \varphi$: So let φ be an $\mathcal{U}$-solution, and $\varphi' := \Phi\varphi$. Then

$$\begin{aligned} u(\Phi'\varphi') &= u\big((\rho \circ \varphi')\restriction Q_\exists\big) \\ &= u\rho\varphi' \\ &= u\varphi, && \text{as proved in 2.} \end{aligned}$$

For every other variable v in $Q_\exists$ we obtain

$$\begin{aligned} v(\Phi'\varphi') &= v\big((\rho \circ \varphi')\restriction Q_\exists\big) \\ &= v\rho\varphi' \\ &= v\varphi' \\ &= v\varphi. \end{aligned}$$

Case flex-flex with different heads, i.e., $\mathcal{U}$ is $Q.u\boldsymbol{y} = v\boldsymbol{z} \wedge C$. Let $\boldsymbol{w}$ be an enumeration of the variables both in $\boldsymbol{y}$ and in $\boldsymbol{z}$. Then $\rho = [u, v := \lambda\boldsymbol{y}.u'\boldsymbol{w}, \lambda\boldsymbol{z}.u'\boldsymbol{w}]$, Q' is Q with $\exists u, \exists v$ removed and $\exists u'$ inserted, and $\mathcal{U}' = Q'C\rho$.

1. Φ' is well-defined: Let φ' be a $\mathcal{U}'$-solution, i.e., assume that $C\rho\varphi'$ holds. We must show that $\varphi := (\rho \circ \varphi')\restriction Q_\exists$ is a $\mathcal{U}$-solution.

For $u\boldsymbol{y} = v\boldsymbol{z}$: We need to show $(u\varphi)\boldsymbol{y} = (v\varphi)\boldsymbol{z}$. But $(u\varphi)\boldsymbol{y} = (u\rho\varphi')\boldsymbol{y} = (\lambda\boldsymbol{y}.(u'\varphi')\boldsymbol{w})\boldsymbol{y} = (u'\varphi')\boldsymbol{w}$, and similarly $(v\varphi)\boldsymbol{z} = (u'\varphi')\boldsymbol{w}$.

For $(r = s) \in C$: We need to show $(r = s)\varphi$. But since u' is a new variable, φ and $\rho \circ \varphi'$ coincide on all variables free in $r = s$, and we have $(r = s)\rho\varphi'$ by assumption.

2. Definition of Φ: $\mathcal{U}$-solutions $\to$ $\mathcal{U}'$-solutions. Let a Q-substitution φ be given such that $(u\boldsymbol{y} = v\boldsymbol{z})\varphi$ and $C\varphi$. Define

$$\begin{aligned} u'(\Phi\varphi) &:= \lambda\boldsymbol{w}.(u\varphi)\boldsymbol{w}\boldsymbol{0} && \text{w.l.o.g.; } \boldsymbol{0} \text{ arbitrary} \\ v'(\Phi\varphi) &:= \lambda\boldsymbol{w}.(v\varphi)\boldsymbol{0}\boldsymbol{w} \\ w(\Phi\varphi) &:= w\varphi && \text{otherwise, i.e., } w \neq u', v', u \text{ flexible.} \end{aligned}$$

Since by assumption $(u\varphi)\boldsymbol{y} = (v\varphi)\boldsymbol{z}$, the normal forms of both $(u\varphi)\boldsymbol{y}$ and $(v\varphi)\boldsymbol{z}$ can only contain the common variables $\boldsymbol{w}$ from $\boldsymbol{y}, \boldsymbol{z}$ free. Hence, for $\varphi' := \Phi\varphi$, $u\rho\varphi' = u\varphi$ by the argument in the previous case, and similarly $v\rho\varphi' = v\varphi$. Since $r\varphi = s\varphi$ ($(r = s) \in C$ arbitrary) by assumption, and ρ only affects u and v, we obtain $r\rho\varphi' = s\rho\varphi'$, as required. $\Phi'(\Phi\varphi) = \varphi$ can now be proved as in the previous case.

Case flex-rigid, $\mathcal{U}$ is $Q.u\boldsymbol{y} = t \wedge C$.

Subcase occurrence check: t contains (a critical subterm with head) u. Then clearly there is no Q-substitution φ such that $(u\varphi)\boldsymbol{y} = t\varphi$.

Subcase pruning: Here t contains a subterm $v\boldsymbol{w}_1 z\boldsymbol{w}_2$ with $\exists v$ in Q, and z free in t. Then $\rho = [v := \lambda\boldsymbol{w}_1, z, \boldsymbol{w}_2.v'\boldsymbol{w}_1\boldsymbol{w}_2]$, Q' is Q with $\exists v$ replaced by $\exists v'$, and $\mathcal{U}' = Q'.u\boldsymbol{y} = t\rho \wedge C\rho$.

1. Φ' is well-defined: Let φ' be a $\mathcal{U}'$-solution, i.e., $(u\varphi')\boldsymbol{y} = t\rho\varphi'$, and $r\rho\varphi' = s\rho\varphi'$ for $(r = s) \in C$. We must show that $\varphi := (\rho \circ \varphi')\restriction Q_\exists$ is a $\mathcal{U}$-solution.

For $u\boldsymbol{y} = t$: We need to show $(u\varphi)\boldsymbol{y} = t\rho\varphi'$. But

$$\begin{aligned} (u\varphi)\boldsymbol{y} &= (u\rho\varphi')\boldsymbol{y} \\ &= (u\varphi')\boldsymbol{y} && \text{since } \rho \text{ does not touch } u \\ &= t\rho\varphi' && \text{by assumption.} \end{aligned}$$

For $(r = s) \in C$: We need to show $(r = s)\varphi$. But since v' is a new variable, $\varphi = (\rho \circ \varphi')\restriction Q_\exists$ and $\rho \circ \varphi'$ coincide on all variables free in $r = s$, and the claim follows from $(r = s)\rho\varphi'$.

2. Definition of $\Phi\colon \mathcal{U}$-solutions $\to \mathcal{U}'$-solutions. For a $\mathcal{U}$-solution φ define

$$\begin{aligned}
v'(\Phi\varphi) &:= \lambda \boldsymbol{w}_1, \boldsymbol{w}_2.(v\varphi)\boldsymbol{w}_1 0 \boldsymbol{w}_2 \\
w(\Phi\varphi) &:= w\varphi \qquad\qquad \text{otherwise, i.e., } w \neq v', v \text{ flexible.}
\end{aligned}$$

Since by assumption $(u\varphi)\boldsymbol{y} = t\varphi$, the normal form of $t\varphi$ cannot contain z free. Therefore, for $\varphi' := \Phi\varphi$,

$$\begin{aligned}
v\rho\varphi' &= (\lambda \boldsymbol{w}_1, z, \boldsymbol{w}_2.v'\boldsymbol{w}_1\boldsymbol{w}_2)\varphi' \\
&= \lambda \boldsymbol{w}_1, z, \boldsymbol{w}_2.(\lambda \boldsymbol{w}_1, \boldsymbol{w}_2.(v\varphi)\boldsymbol{w}_1 0 \boldsymbol{w}_2)\boldsymbol{w}_1\boldsymbol{w}_2 \\
&= \lambda \boldsymbol{w}_1, z, \boldsymbol{w}_2.(v\varphi)\boldsymbol{w}_1 0 \boldsymbol{w}_2 \\
&= \lambda \boldsymbol{w}_1, z, \boldsymbol{w}_2.(v\varphi)\boldsymbol{w}_1 z \boldsymbol{w}_2 \\
&= v\varphi.
\end{aligned}$$

Hence $\varphi' = \Phi\varphi$ satisfies $(u\varphi')\boldsymbol{y} = t\rho\varphi'$. For $r = s$ this follows by the same argument. $\Phi'(\Phi\varphi) = \varphi$ can again be proved as in the previous case.

Subcase pruning impossible: Then $\lambda\boldsymbol{y}t$ has an occurrence of a universally quantified (i.e., forbidden) variable z. Therefore clearly there is no Q-substitution φ such that $(u\varphi)\boldsymbol{y} = t\varphi$.

Subcase explicit definition. Then $\rho = [u := \lambda\boldsymbol{y}t]$, Q' is obtained from Q by removing $\exists u$, and $\mathcal{U}' = Q'C\rho$. Note that ρ is a Q'-substitution, for we have performed the pruning steps.

1. Φ' is well-defined: Let φ' be a $\mathcal{U}'$-solution, i.e., $r\rho\varphi' = s\rho\varphi'$ for $(r = s) \in C$. We must show that $\varphi := (\rho \circ \varphi')\restriction Q_\exists$ is an $\mathcal{U}$-solution.

For $u\boldsymbol{y} = t$: We need to show $(u\rho\varphi')\boldsymbol{y} = t\rho\varphi'$. But

$$\begin{aligned}
(u\rho\varphi')\boldsymbol{y} &= ((\lambda\boldsymbol{y}t)\varphi')\boldsymbol{y} \\
&= t\varphi' \\
&= t\rho\varphi' \qquad\qquad \text{since } u \text{ does not appear in } t.
\end{aligned}$$

For $(r = s) \in C$: We need to show $(r = s)\varphi$. But this clearly follows from $(r = s)\rho\varphi'$.

2. Definition of $\Phi\colon \mathcal{U}$-solutions $\to \mathcal{U}'$-solutions, and proof of $\Phi'(\Phi\varphi) = \varphi$. For a $\mathcal{U}$-solution φ define $\Phi\varphi = \varphi\restriction Q_\exists$. Then

$$u\rho\varphi' = \lambda\boldsymbol{y}t\varphi' = \lambda\boldsymbol{y}t\varphi = u\varphi,$$

and clearly $v\rho\varphi' = v\varphi$ for all other flexible φ. For $(r = s) \in C$, from $r\varphi = s\varphi$ we easily obtain $r\varphi' = s\varphi'$.

It is not hard to see that the unification algorithm terminates, by defining a measure that decreases with each transition.

Corollary 2. *Given a unification problem $\mathcal{U} = QC$, the unification algorithm either returns $\#f$, and there is no $\mathcal{U}$-solution, or else returns a pair (Q', ρ) with a "transition" substitution ρ and a prefix Q' (i.e., a unification problem $\mathcal{U}'$*

with no unification pairs) such that for any Q'-substitution φ', $(\rho \circ \varphi')\restriction Q_\exists$ is an $\mathcal{U}$-solution, and every $\mathcal{U}$-solution can be obtained in this way. Since the empty substitution is a Q'-substitution, $\rho\restriction Q_\exists$ is an $\mathcal{U}$-solution, which is most general in the sense stated.

$\mathsf{unif}(Q, \boldsymbol{r} = \boldsymbol{s})$ denotes the result of the unification algorithm at $\mathcal{U} = Q\boldsymbol{r} = \boldsymbol{s}$.

4 Proof search

A *Q-sequent* has the form $\mathcal{P} \Rightarrow G$, where $\mathcal{P}$ is a list of Q-clauses and G is a Q-goal.

We write $M[\mathcal{P}]$ to indicate that all assumption variables in the derivation M concern clauses in $\mathcal{P}$.

Write $\vdash^n S$ for a set S of sequents if there are derivations $M_i^{G_i}[\mathcal{P}_i]$ in long normal form for all $(\mathcal{P}_i \Rightarrow G_i) \in S$ such that $\sum \#M_i \leq n$. Let $\vdash^{<n} S$ mean $\exists m{<}n \vdash^m S$.

We now prove correctness and completeness of the proof search procedure: correctness is the if-part of the two lemmata to follow, and completeness the only-if-part.

Lemma 3. *Let Q be a $\forall\exists\forall$-prefix, $\{\mathcal{P} \Rightarrow \forall \boldsymbol{x}.\boldsymbol{D} \to A\} \cup S$ Q-sequents with $\boldsymbol{x}, \boldsymbol{D}$ not both empty. Then we have for every substitution φ:*

$$\varphi \textit{ is a } Q\textit{-substitution such that} \vdash^n \big(\{\mathcal{P} \Rightarrow \forall \boldsymbol{x}.\boldsymbol{D} \to A\} \cup S\big)\varphi$$

if and only if

$$\varphi \textit{ is a } Q\forall\boldsymbol{x}\textit{-substitution such that} \vdash^{<n} \big(\{\mathcal{P} \cup \boldsymbol{D} \Rightarrow A\} \cup S\big)\varphi.$$

Proof. *"If"*. Let φ be a $Q\forall\boldsymbol{x}$-substitution and $\vdash^{<n} \big(\{\mathcal{P} \cup \boldsymbol{D} \Rightarrow A\} \cup S\big)\varphi$. So we have

$$N^{A\varphi}[\boldsymbol{D}\varphi \cup \mathcal{P}\varphi].$$

Since φ is a $Q\forall\boldsymbol{x}$-substitution, no variable in $\boldsymbol{x}$ can be free in $\mathcal{P}\varphi$, or free in $y\varphi$ for some $y \in \mathsf{dom}(\varphi)$. Hence

$$M^{(\forall \boldsymbol{x}.\boldsymbol{D} \to A)\varphi}[\mathcal{P}\varphi] := \lambda \boldsymbol{x} \lambda \boldsymbol{u}^{\boldsymbol{D}\varphi} N$$

is a correct derivation.

"Only if". Let φ be a Q-substitution and $\vdash^n \big(\{\mathcal{P} \Rightarrow \forall \boldsymbol{x}.\boldsymbol{D} \to A\} \cup S\big)\varphi$. This means we have a derivation (in long normal form)

$$M^{(\forall \boldsymbol{x}.\boldsymbol{D} \to A)\varphi}[\mathcal{P}\varphi] = \lambda \boldsymbol{x} \lambda \boldsymbol{u}^{\boldsymbol{D}\varphi}.N^{A\varphi}[\boldsymbol{D}\varphi \cup \mathcal{P}\varphi].$$

Now $\#N < \#M$, hence $\vdash^{<n} \big(\{\mathcal{P} \cup \boldsymbol{D} \Rightarrow A\} \cup S\big)\varphi$, and φ clearly is a $Q\forall\boldsymbol{x}$-substitution.

Lemma 4. *Let Q be a $\forall\exists\forall$-prefix, $\{\mathcal{P} \Rightarrow P\boldsymbol{r}\} \cup S$ Q-sequents and φ a substitution. Then*

$$\varphi \text{ is a } Q\text{-substitution such that } \vdash^n \big(\{\mathcal{P} \Rightarrow P\boldsymbol{r}\} \cup S\big)\varphi$$

if and only if there is a clause $\forall \boldsymbol{x}.\boldsymbol{G} \to P\boldsymbol{s}$ in $\mathcal{P}$ such that the following holds. Let $\boldsymbol{z}$ be the final universal variables in Q, $\boldsymbol{x}$ be new ("raised") variables such that $X_i\boldsymbol{z}$ has the same type as x_i, let Q^* be Q with the existential variables extended by $\boldsymbol{x}$, and let $*$ indicate the substitution $[x_1, \dots, x_n := X_1\boldsymbol{z}, \dots, X_n\boldsymbol{z}]$. Then $\mathsf{unif}(Q^*, \boldsymbol{r} = \boldsymbol{s}^*) = (Q', \rho)$ and there is a Q'-substitution φ' such that $\vdash^{<n} \big(\{\mathcal{P} \Rightarrow \boldsymbol{G}^*\} \cup S\big)\rho\varphi'$, and $\varphi = (\rho \circ \varphi')\restriction Q_\exists$.

Proof. *"If".* Let $\mathsf{unif}(Q^*, \boldsymbol{r} = \boldsymbol{s}^*) = (Q', \rho)$, and assume that φ' is a Q'-substitution such that $N_i \vdash \big(\mathcal{P} \Rightarrow \boldsymbol{G}^*\big)\rho\varphi'$. Let $\varphi := (\rho \circ \varphi')\restriction Q_\exists$. From $\mathsf{unif}(Q^*, \boldsymbol{r} = \boldsymbol{s}^*) = (Q', \rho)$ we know $\boldsymbol{r}\rho = \boldsymbol{s}^*\rho$, hence $\boldsymbol{r}\varphi = \boldsymbol{s}^*\rho\varphi'$. Then

$$u^{(\forall \boldsymbol{x}.\boldsymbol{G} \to P\boldsymbol{s})\varphi}\big((\boldsymbol{x}\rho\varphi')\boldsymbol{z}\big)\boldsymbol{N}^{\boldsymbol{G}^*\rho\varphi'}$$

derives $P\boldsymbol{s}^*\rho\varphi'$ (i.e., $P\boldsymbol{r}\varphi$) from $\mathcal{P}\varphi$.

"Only if". Assume φ is a Q-substitution such that $\vdash (\mathcal{P} \Rightarrow P\boldsymbol{r})\varphi$, say by $u^{(\forall \boldsymbol{x}.\boldsymbol{G} \to P\boldsymbol{s})\varphi}\boldsymbol{t}\boldsymbol{N}^{(\boldsymbol{G}\varphi)[\boldsymbol{x}:=\boldsymbol{t}]}$, with $\forall \boldsymbol{x}.\boldsymbol{G} \to P\boldsymbol{s}$ a clause in $\mathcal{P}$, and with additional assumptions from $\mathcal{P}\varphi$ in $\boldsymbol{N}$. Then $\boldsymbol{r}\varphi = (\boldsymbol{s}\varphi)[\boldsymbol{x} := \boldsymbol{t}]$. Since we can assume that the variables $\boldsymbol{x}$ are new and in particular not range variables of φ, with

$$\vartheta := \varphi \cup [\boldsymbol{x} := \boldsymbol{t}]$$

we have $\boldsymbol{r}\varphi = \boldsymbol{s}\vartheta$. Let $\boldsymbol{z}$ be the final universal variables in Q, $\boldsymbol{x}$ be new ("raised") variables such that $X_i\boldsymbol{z}$ has the same type as x_i, let Q^* be Q with the existential variables extended by $\boldsymbol{x}$, and for terms and formulas let $*$ indicate the substitution $[x_1, \dots, x_n := X_1\boldsymbol{z}, \dots, X_n\boldsymbol{z}]$. Moreover, let

$$\vartheta^* := \varphi \cup [X_1, \dots, X_n := \lambda \boldsymbol{z}.t_1, \dots, \lambda \boldsymbol{z}.t_n].$$

Then $\boldsymbol{r}\vartheta^* = \boldsymbol{r}\varphi = \boldsymbol{s}\vartheta = \boldsymbol{s}^*\vartheta^*$, i.e., ϑ^* is a solution to the unification problem given by Q^* and $\boldsymbol{r} = \boldsymbol{s}$. Hence by Lemma 2 $\mathsf{unif}(Q^*, \boldsymbol{r} = \boldsymbol{s}^*) = (Q', \rho)$ and there is a Q'-substitution φ' such that $\vartheta^* = (\rho \circ \varphi')\restriction Q^*_\exists$, hence $\varphi = (\rho \circ \varphi')\restriction Q_\exists$. Also, $(\boldsymbol{G}\varphi)[\boldsymbol{x} := \boldsymbol{t}] = \boldsymbol{G}\vartheta = \boldsymbol{G}^*\vartheta^* = \boldsymbol{G}\rho\varphi'$.

A *state* is a pair (Q, S) with Q a prefix and S a finite set of Q-sequents. By the two lemmas just proved we have *state transitions*

$$(Q, \{\mathcal{P} \Rightarrow \forall \boldsymbol{x}.\boldsymbol{D} \to A\} \cup S) \mapsto^{\varepsilon} (Q\forall \boldsymbol{x}, \{\mathcal{P} \cup \boldsymbol{D} \Rightarrow A\} \cup S)$$
$$(Q, \{\mathcal{P} \Rightarrow P\boldsymbol{r}\} \cup S) \mapsto^{\rho} (Q', (\{\mathcal{P} \Rightarrow \boldsymbol{G}^*\} \cup S)\rho),$$

where in the latter case there is a clause $\forall \boldsymbol{x}.\boldsymbol{G} \to P\boldsymbol{s}$ in $\mathcal{P}$ such that the following holds. Let $\boldsymbol{z}$ be the final universal variables in Q, $\boldsymbol{x}$ be new ("raised") variables such that $X_i\boldsymbol{z}$ has the same type as x_i, let Q^* be Q with the existential variables extended by $\boldsymbol{x}$, and let $*$ indicate the substitution $[x_1, \dots, x_n := X_1\boldsymbol{z}, \dots, X_n\boldsymbol{z}]$, and $\mathsf{unif}(Q^*, \boldsymbol{r} = \boldsymbol{s}^*) = (Q', \rho)$.

Notice that by Lemma 1, if $\mathcal{P} \Rightarrow P\boldsymbol{r}$ is a *Q-sequent* (which means that $\bigwedge\!\!\!\bigwedge \mathcal{P} \to P\boldsymbol{r}$ is a Q-goal), then $(\mathcal{P} \Rightarrow \boldsymbol{G}^*)\rho$ is a Q'-sequent.

Theorem 1. *Let Q be a prefix, and S be a set of Q-sequents. For every substitution φ we have: φ is a Q-substitution satisfying $\vdash S\varphi$ iff there is a prefix Q', a substitution ρ and a Q'-substitution φ' such that*

$$(Q, S) \mapsto^{\rho *} (Q', \emptyset),$$
$$\varphi = (\rho \circ \varphi') \restriction Q_\exists.$$

Examples. 1. The sequent $\forall y.\forall z Ryz \to Q, \forall y_1, y_2 R y_1 y_2 \Rightarrow Q$ leads first to $\forall y_1, y_2 R y_1 y_2 \Rightarrow Ryz$ under $\exists y \forall z$, then to $y_1 = y \wedge y_2 = z$ under $\exists y \forall z \exists y_1, y_2$, and finally to $Y_1 z = y \wedge Y_2 z = z$ under $\exists y, Y_1, Y_2 \forall z$, which has the solution $Y_1 = \lambda z y$, $Y_2 = \lambda z z$.

2. $\forall y.\forall z Ryz \to Q, \forall y_1 R y_1 y_1 \Rightarrow Q$ leads first to $\forall y_1 1 R y_1 y_1 \Rightarrow Ryz$ under $\exists y \forall z$, then to $y_1 = y \wedge y_1 = z$ under $\exists y \forall z \exists y_1$, and finally to $Y_1 z = y \wedge Y_1 z = z$ under $\exists y, Y_1 \forall z$, which has no solution.

3. Here is a more complex example (derived from proofs of the Orevkov-formulas), for which we only give the derivation tree.

$$\dfrac{\dfrac{\forall y.(\forall z Ryz \to \bot) \to \bot}{(\forall z R0z \to \bot) \to \bot} \qquad \dfrac{\dfrac{\dfrac{\dfrac{\forall y.(\forall z_1 Ryz_1 \to \bot) \to \bot}{(\forall z_1 Rzz_1 \to \bot) \to \bot} \qquad \dfrac{\dfrac{\dfrac{\dfrac{\forall z S0z \to \bot}{S0z_1 \to \bot} \quad \dfrac{(*) \quad R0z \quad Rzz_1}{S0z_1}}{\bot}}{Rzz_1 \to \bot}}{\forall z_1 Rzz_1 \to \bot}}{\bot}}{R0z \to \bot}}{\forall z R0z \to \bot}}{\bot}$$

where $(*)$ is a derivation from $\mathrm{Hyp}_1\colon \forall z, z_1. R0z \to Rzz_1 \to S0z_1$.

5 Extension by $\wedge$ and $\exists$

The extension by conjunction is rather easy; it is even superfluous in principle, since conjunctions can always be avoided at the expense of having lists of formulas instead of single formulas.

However, having conjunctions available is clearly useful at times, so let's add it. This requires the notion of an *elaboration path* for a formula (cf. [1]). The reason is that the property of a formula to have a unique atom as its *head* is lost when conjunctions are present. An elaboration path is meant to give the directions (left or right) to go when we encounter a conjunction as a strictly positive subformula. For example, the elaboration paths of $\forall x A \wedge (B \wedge C \to D \wedge \forall y E)$ are (`left`), (`right`, `left`) and (`right`, `right`). Clearly, a formula is equivalent to the conjunction (over all elaboration paths) of all formulas obtained from it by following an elaboration path (i.e., always throwing away the other part of the conjunction). In our example,

$$\forall x A \wedge (B \wedge C \to D \wedge \forall y E) \leftrightarrow \forall x A \wedge (B \wedge C \to D) \wedge (B \wedge C \to \forall y E).$$

In this way we regain the property of a formula to have a unique head, and our previous search procedure continues to work.

For the existential quantifier $\exists$ the problem is of a different nature. We chose to introduce $\exists$ by means of axiom schemata. Then the problem is which of such schemes to use in proof search, given a goal G and a set $\mathcal{P}$ of clauses. We might proceed as follows.

List all prime, positive and negative existential subformulas of $\mathcal{P} \Rightarrow G$, and remove any formula from those lists which is of the form of another one[1]. For every positive existential formula – say $\exists x B$ – add (the generalization of) the existence introduction scheme

$$\exists^{+}_{x,B} \colon \forall x.B \to \exists x B$$

to $\mathcal{P}$. Moreover, for every negative existential formula – say $\exists x A$ – and every (prime or existential) formula C in any of those two lists, exept the formula $\exists x A$ itself, add (the generalization of) the existence elimination scheme

$$\exists^{-}_{x,A,C} \colon \exists x A \to (\forall x.A \to C) \to C$$

to $\mathcal{P}$. Then start the search algorithm as described in section 4. The normal form theorem for the natural deduction system of minimal logic with $\exists$ then guarantees completeness.

However, experience has shown that this complete search procedure tends to be trapped in too large a search space. Therefore in our actual implementation we decided to only take instances of the existence elimination scheme with *existential* conclusions.

Acknowledgements I have benefitted from a presentation of Miller's [1] given by Ulrich Berger, in a logic seminar in München in June 1991.

References

1. Dale Miller. A logic programming language with lambda–abstraction, function variables and simple unification. *Journal of Logic and Computation*, 2(4):497–536, 1991.
2. Tobias Nipkow. Higher-order critical pairs. In R. Vemuri, editor, *Proceedings of the Sixth Annual IEEE Symposium on Logic in Computer Science*, pages 342–349, Los Alamitos, 1991. IEEE Computer Society Press.

[1] To do this, for patterns the dual of the theory of "most general unifiers", i.e., a theory of "most special generalizations", needs to be developed.