

A Many-Sorted Calculus with Polymorphic Functions Based on Resolution and Paramodulation.

Manfred Schmidt-Schauss
Fachbereich Informatik
Universität Kaiserslautern
6750 Kaiserslautern, Germany

ABSTRACT.

A many-sorted first order calculus, called ERP [Wa83], whose well formed formulas are typed clauses and whose inference rules are factorization, resolution, paramodulation and weakening is extended to a many sorted calculus ERP^{*} with polymorphic functions (overloading). It is assumed that the sort structure is a finite partially ordered set with a greatest element. It is shown, that this extended calculus is sound and complete, provided the functional reflexivity axioms are present. It is also shown, that unification of terms containing polymorphic functions is in general finitary, i.e. the set of most general unifiers may contain more than one element, but at most finitely many.

We give a natural condition for the signature (the sort structure), such that the set of most general unifiers is always at most a singleton provided this condition holds.

1. Introduction.

The advantages of a many-sorted calculus in automated reasoning systems are well known [Hay71, Hen72, Wa83, GM84, GM85, Co83, CD83, Ob62], a fact that was also noticed in logical programming [MO84, Mi84]:

In a many-sorted calculus we obtain a shorter refutation of a smaller set of shorter clauses, as compared to the unsorted version.

Our interest is in using a many sorted calculus in Automated Theorem Proving, especially in resolution based ATP-systems. The most desirable properties of such a calculus are:

- 1.) The unsorted problem and the corresponding sorted problem are equivalent, i.e. the unsorted clause set is contradictory, iff the sorted version is contradictory.
- 2.) The calculus is complete, i.e. there is a derivation of the empty clause, iff the clause set is contradictory.

- 3.) The search space in the sorted version of a problem is smaller than the search space of the unsorted version (provided the problem has a sort structure).
- 4.) The many-sorted calculus has as much expressive power as possible.
- 5.) The calculus should be based on standard resolution and paramodulation [WR73] possibly augmented by a modified unification algorithm. Hence standard reductions like purity, subsumption, tautology deletion, replacement resolution and incompatibility of unifiers [KM84, Ro64] can still be used. Furthermore resolution based strategies such as Unit resolution and Set-of-support should be applicable.

The ERP-calculus of C. Walther [Wa83] essentially satisfies these requirements, but can be improved by the additional incorporation of polymorphic functions (overloading). This new extended calculus is called ERP^{*}.

In ERP^{*} it is possible for example to have a function symbol + (sum) denoting the addition of (complex) numbers with the implicitly stated property, that, syntactically, the sum of integers is an integer, the sum of reals is a real and the sum of Gaussian numbers is a Gaussian number. However we do not allow to use the same symbol "+" for say the addition of vectors, since we have the technical restriction, that for every argument position in every function there exists a greatest sort for that argument. Using "+" for numbers and vectors would imply that e.g. $1+(0,0)$ is well sorted, which does not make sense.

Without this restriction, the ERP^{*}-calculus is not complete in general. (This however could be remedied if either ill-sorted terms are allowed during equality deductions or an extended parallel paramodulation rule is used).

The results presented in this paper concern unification of polymorphic terms and completeness of the ERP^{*}-calculus. In particular it is shown, that the complete and minimal set of most general unifiers

for two polymorphic terms is always finite.

Some authors present sort structures, such that the union, the intersection and the complement of sorts are defined [Co83, CD83]. If such information is used in a deduction, the rules of the ERP⁺-calculus are not sufficient and extra rules would be necessary to ensure completeness. But such extra rules have in general the very unpleasant side effect that the reductions and strategies of a resolution based calculus are no longer applicable.

2. Basic Notions of the ERP⁺-calculus.

In the following we specify the underlying logic. The non standard part of the next definitions are the polymorphic functions. We assign to every function the set of all domain-range relations, e.g. for the sum (+) of complex numbers we have:

$+: \text{COMPLEX} \times \text{COMPLEX} \rightarrow \text{COMPLEX};$
 $\text{COMPLEX} \times \text{INT} \rightarrow \text{COMPLEX};$
 $\text{INT} \times \text{INT} \rightarrow \text{INT}; \dots$

This is denoted as a set of triples
 $\{(\text{COMPLEX}, \text{COMPLEX}, \text{COMPLEX}), (\text{INT}, \text{INT}, \text{INT}), (\text{COMPLEX}, \text{INT}, \text{COMPLEX}), \dots\}.$

Definition: A polymorphic signature SIG is a triple (S, F, P) , where

- 1) S is the finite set of sorts, $\leq$ is a partial ordering on S with the greatest element τ . $\leq$ is extended to tuples of sorts in the usual way (componentwise $\leq$).
- 2) F is the set of function symbols. $F = \bigcup F_W$, where F_W is the set of function symbols of arity n with $\emptyset \neq W \subseteq S^{n+1}$.
 If $F_W \neq \emptyset$, then W satisfies the following conditions:
 - The sort of constants is unique. i.e. $W \subseteq S$ implies $|W| = 1$.
 - For signatures of functions W , which are not constant:
 - a) W contains a unique greatest element $(S_{W,1}, S_{W,2}, \dots, S_{W,n+1})$.
 - b) For every $(S_1, S_2, \dots, S_{n+1}) \in W$ and every $(T_1, \dots, T_n) \in S^n$, $(T_1, \dots, T_n) \leq (S_1, \dots, S_n)$ implies that there exists a unique sort $T_{n+1} \leq S_{n+1}$ such that $(T_1, \dots, T_{n+1}) \in W$. That means that for every $f \in F$ the related function $f^*: ((S_1, \dots, S_n) \mid (S_1, \dots, S_n) \leq (S_{W,1}, \dots, S_{W,n})) \rightarrow S$, where $f^*(S_1, \dots, S_n) = S_{n+1}$, iff $(S_1, \dots, S_{n+1}) \in W$, is well defined and monotone.

3) P is the set of predicate symbols. If P_D is the set of predicates with domain D , where $D \in S^n$. We define $P = \bigcup P_D$.

4) For every sort $S \in S$, there exists a constant c of sort $S' \leq S$. That means, that SIG is strict in the sense of [HO80].

We use the following additional notation and abbreviations: $SO(f) = W$, iff $f \in F_W$; $SO(P) = V$, iff

$P \in P_V$. C denotes the set of all constants, C_S denotes the set of all constants of sort S .

If the set of ranges of a function f , i.e. the set $\{S_{n+1} \mid (S_1, \dots, S_{n+1}) \in SO(f)\}$ has more than one element, then f is called a polymorphic function. If no polymorphic function is in F , then SIG corresponds to a signature of the ERP-calculus [Wa83].

In an actual implementation, it is not necessary, that the whole signature of a function is explicitly specified. It is sufficient to give enough information to compute the signature of a function uniquely. For functions, which are not polymorphic, the specification of the maximal domain and maximal range suffices.

Let V_S be the (infinite) set of variables of sort S and

let $V = \bigcup V_S$. Terms (including ill-sorted terms) are defined as usual. The set of all terms is called TERM. The sort of t , namely $[t]$, is defined as follows:

The function $[...]$ maps constants and variables to their respective sorts and is recursively extended to terms by $[f(t_1, \dots, t_n)] = S_{n+1}$, iff $S_i = [t_i]$ and

$(S_1, \dots, S_{n+1}) \in SO(f)$. We define ΣTERM , the set of all well-sorted terms as the domain of the function $[...]$, i.e. the terms, for which a sort is defined.

The functional reflexivity axioms are the axioms

$\forall x \ x = x$, and $\forall x_1, \dots, x_n \ f(x_1, \dots, x_n) = f(x_1, \dots, x_n)$.

Example. Let $S = \{N, NZ, Z\}$, where N denotes the nonnegative integers, including 0, NZ denotes the positive integers and $Z = \{0\}$. Then $N \geq NZ$ and $N \geq Z$. The function $+$ has the following signature:

$SO(+)$ = $\{(N, N, N); (N, NZ, NZ); (NZ, N, NZ); (Z, N, N); (NZ, N, N); (Z, NZ, NZ); (NZ, Z, NZ); (Z, Z, Z); (NZ, NZ, NZ)\}$.

Then for example $[0+0] = Z$ and $[0+1] = NZ$.

Definition. For a given signature $SIG = (S, F, P)$ we say the triple (A, S, F) is an algebra of type (S, F) , iff the following is satisfied:

- A is a nonempty set.
- To every sort $S \in \mathcal{S}$ is assigned a subset S^A of A such that $\tau^A = A$ and for all $S, T \in \mathcal{S}$: $S \leq T \Rightarrow S^A \subseteq T^A$.
- For $c \in C_S$, an element $c^A \in A$ exists, such that $c^A \in S^A$.
- For functions f , $f^A: A^n \rightarrow A$ is a mapping such that the image of $S_1^A \times \dots \times S_n^A$ is in S_{n+1}^A for all $(S_1, \dots, S_{n+1}) \in SO(f)$. $\square$

By the definition of a signature, we have that $S^A \neq \emptyset$ for every $S \in \mathcal{S}$ since every sort contains at least one constant.

Definition. A mapping $\varphi: A \rightarrow B$, which has the properties, that

- a) $\varphi(S^A) \subseteq \varphi(S^B)$ for every sort $S \in \mathcal{S}$ and
- b) $\varphi(f^A(a_1, \dots, a_n)) = f^B(\varphi(a_1), \dots, \varphi(a_n))$

is called a Σ -homomorphism. $\square$

$(\Sigma\text{TERM}, S, F)$ is the free algebra of type (S, F) and $(\Sigma\text{TERM}_{gr}, S, F)$ is the initial algebra of type (S, F) , where ΣTERM_{gr} is the set of terms without variables.

A mapping $\sigma: \Sigma\text{TERM} \rightarrow \Sigma\text{TERM}$, which is a Σ -endomorphism where $\{x \in V \mid \sigma x = x\}$ is finite, is called a Σ -substitution. Let ΣSUB be the set of all Σ -substitutions. Σ -substitutions are exactly those mappings, which are substitutions in the normal sense and have the additional property that $\sigma(x) \leq [x]$ for all $x \in V$.

$P(t_1, \dots, t_n)$ is an atom, where P is a predicate symbol and the t_i 's are terms such that $[t_i] \leq S_i$ where $(S_1, \dots, S_n) = SO(P)$. A literal is a signed atom. A clause is a set of literals, i.e. an abbreviation for the disjunction of the literals, where all variables are universally quantified. A ground atom, a ground literal or a ground clause is one without variables. Instances of atoms, literals and clauses are their images under a Σ -substitution. Equality ($=$) is a distinguished binary predicate with domainsorts $SO(=) = (\tau, \tau)$.

Since a clause set CS is said to be satisfiable, if and only if a model for CS exists, there is the need for a precise definition of a model respecting polymorphic signatures.

Definition. A ΣE -model for a clause set CS is a triple (D, SIG, R) , which satisfies the following conditions:

- (D, S, F) is an algebra of type (S, F) .
- For every predicate P there exists a relation $P^D \in R$ of the same arity.
- All clauses in CS are valid under all Σ -homomorphisms $\varphi: \Sigma\text{TERM} \rightarrow D$, i.e. all clauses are valid under all assignments of values in D to variables in clauses, where sorts are respected. (A literal $+P(\dots)$ is valid under φ , if its image under φ is in the relation P^D).
- The equality predicate is represented as the identity on D .

3. Unification of Polymorphic Terms.

For $\sigma \in \Sigma\text{SUB}$ we use the notation:

$\text{DOM}(\sigma) = \{x \in V \mid \sigma x \neq x\}$; $\text{COD}(\sigma) = \{\sigma x \mid x \in \text{DOM}(\sigma)\}$ and $\text{VCOD}(\sigma)$ is the set of variables in $\text{COD}(\sigma)$. A unification problem for two terms s and t is denoted as $\langle s = t \rangle$. The set of variables of a term t is denoted as $\text{VAR}(t)$.

For $W \subseteq V$ and $\sigma, \tau \in \Sigma\text{SUB}$, define $\sigma = \tau [W]$, iff $\sigma x = \tau x$ for all $x \in W$. Furthermore define $\sigma \leq \tau [W]$, iff $\sigma = \lambda \tau [W]$ for some $\lambda \in \Sigma\text{SUB}$. Obviously $= [W]$ is a reflexive and transitive relation on Σ -substitutions. For a subset $U \subseteq \Sigma\text{SUB}$ we say U is separated on a set of variables W , iff $\text{DOM}(\sigma) = W$ and $\text{VCOD}(\sigma) \cap W = \emptyset$ for all $\sigma \in U$ and all VCOD 's are disjoint. Note that such a set consists of idempotent Σ -substitutions only.

Definition. A set of most general unifiers $\mu\text{UE}(s, t)$ for two terms s and t is defined as a subset of ΣSUB , which is separated on $W = \text{VAR}(s, t)$ and satisfies:

- $\sigma s = \sigma t$ for all $\sigma \in \mu\text{UE}(s, t)$. (correctness).
- For all $\delta \in \Sigma\text{SUB}$ with $\delta s = \delta t$ we have $\delta \leq \sigma [W]$ for some $\sigma \in \mu\text{UE}(s, t)$. (completeness)
- $\sigma \leq \tau [W] \Rightarrow \sigma = \tau$ for all $\sigma, \tau \in \mu\text{UE}(s, t)$. (minimality) $\square$

Definition. A Σ -substitution is a weakening substitution (coercing) [Wa84, GM85], iff

- (i) σ substitutes variables for variables and
- (ii) σ is injective on $\text{DOM}(\sigma)$ and
- (iii) $\text{DOM}(\sigma) \cap \text{VCOD}(\sigma) = \emptyset$. $\square$

The set of all weakening substitutions is called ΣWSUB . In the computation of a set of most general unifiers, weakening substitutions are used to solve unification problems like $\langle x = t \rangle$, where $[t]$ is not a subsort of $[x]$.

Definition. The set of most general weakening substitutions for t and S , $\mu\text{WE}_S(t)$ is a set of weakening substitutions which is separated on $W = \text{VAR}(t)$ and satisfies:

- $[\sigma t] \leq S$ for all $\sigma \in \mu W\Sigma_S(t)$. (correctness)
- $\forall \delta \in \Sigma\text{SUB}$ with $[\delta t] \leq S$ there exists a $\sigma \in \mu W\Sigma_S(t)$ such that $\delta \leq \sigma[W]$. (completeness)
- $\forall \sigma, \tau \in \mu W\Sigma_S(t)$: $\sigma \leq \tau[W] \Rightarrow \sigma = \tau$. (minimality) $\square$

This definition is of course only useful for terms, which are ill-sorted or for terms t where $[t]$ is not a subsort of S and the outermost function symbol of t is polymorphic.

The following theorem is valid in the ΣRP -calculus [Wa83, Wa84] and can be extended to ΣRP^* .

Theorem 1. [Sch85a]. Let $t \in \text{TERM}$ and $S \in \mathcal{S}$. If there exists a $\theta \in \Sigma\text{SUB}$ such that $[\theta t] \leq S$, then $\mu W\Sigma_S(t)$ exists, it is not empty and it is always finite.

An immediate consequence is:

Theorem 2. For unifiable x and t the set $\mu\text{UE}(x, t)$ exists, is not empty and finite. $\square$

The proof follows from the fact, that $\mu\text{UE}(x, t) = \{\sigma(x \leftarrow \sigma t) \mid \sigma \in \mu W\Sigma_{[x]}(t)\}$, if $x \notin \text{VAR}(t)$.

Example. The following example demonstrates, that for a unification problem $\langle x, t \rangle$ the minimal set of most general unifiers can grow exponentially.

Consider the sort structure $S = \{N, NZ, Z\}$, where N , NZ and Z have the same meaning as in the example above. Let $x \in V_{NZ}$ and $x_1 \in V_N$. The signature of the

function $*$ (product) is:

$\text{SO}(*) = \{(N, N, N), (NZ, N, N), (N, NZ, N), (Z, N, Z), (N, Z, Z), (Z, NZ, Z), (NZ, Z, Z), (Z, Z, Z), (NZ, NZ, NZ)\}$. The unification problem $\langle x = (x_1 + x_2) * \dots * (x_{2n-1} + x_{2n}) \rangle$ produces 2^n

unifiers, since for every factor there are two independent solutions $\{x_{2i-1} \leftarrow y_{2i-1}\}$ and $\{x_{2i} \leftarrow y_{2i}\}$, where $[y_i] = NZ$. These solutions have to be combined independently.

For a set of variables W and a set of Σ -substitutions U , the set $\text{MAX}_{[W]}(U)$ is a set of Σ -substitutions, such that

- i) $\text{MAX}_{[W]}(U) \subseteq U$ and
- ii) $\forall \sigma \in \text{MAX}_{[W]}(U), \tau \in U: \sigma \leq \tau[W] \Rightarrow \sigma = \tau$.
- iii) $\forall \tau \in U: \exists \sigma \in \text{MAX}_{[W]}(U): \tau \leq \sigma[W]$.

Such a set exists, since $[W]$ is transitive.

The polymorphic unification algorithm ΣUNIFY , which is an extension of the algorithm for the ΣRP -calculus [Wa84], takes two terms s, t as input and produces a set of most general unifiers as output.

It is defined as follows: [Sch85a]

- 1) $U_{\text{NEW}} = \{\epsilon\}$.
- 2) REPEAT $U_{\text{OLD}} := U_{\text{NEW}}, U_{\text{NEW}} = \emptyset$.

For every $\sigma \in U_{\text{OLD}}$ DO

Let (d, e) be the first disagreement pair of the two terms $\sigma s, \sigma t$.

IF d or e is a variable, THEN DO

$U_{\text{NEW}} = U_{\text{NEW}} \cup \{\sigma \cdot \tau \mid \tau \in \mu\text{UE}(d, e)\}$.

OD.

OD.

UNTIL U_{NEW} is a set of Σ -unifiers for s, t or

$U_{\text{NEW}} = \emptyset$.

- 3) RETURN $\text{MAX}_{[W]}(U_{\text{NEW}})$. $\square$

We have the theorem, which is proved in [Sch85a]:

Theorem 3. Let $s, t \in \Sigma\text{TERM}$ be Σ -unifiable. Then $\Sigma\text{UNIFY}(s, t)$ terminates and returns a finite nonempty set, which is equivalent to $\mu\text{UE}(s, t)$. $\square$

Definition. We say SIG is a unification unique signature, iff

- $\langle S, \leq \rangle$ is a semilattice with a unique greatest element.
- For all $f \in F$ and all $S \in \mathcal{S}$, the set $\{(S_1, \dots, S_{n+1}) \in \text{SO}(f) \mid S_{n+1} \leq S\}$ is either empty or contains a unique greatest element.

The following theorem is proved in [Sch85a].

Theorem 4. Let SIG be a unification unique signature. Then $\mu\text{UE}(s, t)$ is at most a singleton for all $s, t \in \Sigma\text{TERM}$. $\square$

In [Sch85a] it is shown, that the condition, that $\langle S, \leq \rangle$ is a semilattice, does not restrict the expressiveness of the many-sorted calculus ΣRP^* . In the case that $\langle S, \leq \rangle$ is a partially ordered set, it is possible to add sorts to the signature, such that the resulting sort structure is a semilattice and the satisfiability of clause sets is unchanged.

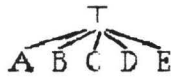
4. The ΣRP^* -calculus.

Σ -factorization, Σ -resolution and Σ -paramodulation are defined as in [Wa83], however based on the unifier sets μUE and μPE respectively. The set μPE is the set of most general unifiers for paramodulation, which is also always finite. The Σ -paramodulation rule includes the weakening rule of [Wa83]. The proofs in [WR73] can be generalized (see [Sch85a]) to the ΣRP^* -calculus yielding the following result:

Theorem 5. Let CS be a clause set containing the functional reflexivity axioms. Then:

$\text{CS} \vdash_{\Sigma\text{RP}^*} \text{FALSE} \iff \text{CS is unsatisfiable.}$

Example. The functional reflexivity axioms are needed for the completeness of the Σ RP-calculus:
Let $S = \{T, A, B, C, D, E\}$.



- Let a, b, c, d be constants of sort A, B, C, D respectively.
- Let P be a unary predicate with $SO(P) = (E)$.
- Let $f \in F$ with a signature, such that $[f(t_1, t_2)] = E$, iff $([t_1] \leq A \text{ and } [t_2] \leq C) \text{ or } ([t_1] \leq B \text{ and } [t_2] \leq D)$; otherwise $[f(t_1, t_2)] = S$.
- Let CS be the clause set $\{ \{a=b\}, \{c=d\}, \{P(f(a, c))\}, \{\neg P(f(b, d))\} \}$.

This clause set is unsatisfiable, but Σ -paramodulation is not possible, since $[f(b, c)] = [f(a, d)] = S$. There is no deduction of the empty clause in ΣRP^* . If the functional reflexivity axioms are present, then $f(a, c) = f(b, d)$ can be deduced from the functional reflexivity axioms and the clauses in CS . Hence the empty clause is then deducible. $\square$

This result is somewhat theoretical. In practical applications, the functional reflexivity axioms are not used, since (i) they increase the search space enormously and (ii) the subsumption rule would delete all of them but the axiom $x=x$. Clearly, this omission may be the reason for incompleteness.

5. The Sort-Theorem. For a polymorphic signature SIG and a related clause set CS , we define an unsorted relativization CS_{rel} [Wa83, Ob62] as follows:

For every sort S use a unary predicate P_S .

The set of sort axioms A^Σ consists of:

- $\forall x \neg P_S(x) \vee P_T(x)$ if $S \leq T$ for $S, T \in S$.
- $\forall x P_S(x)$ for the greatest sort S of S .
- $\forall x_1, \dots, x_n \neg P_{S_1}(x_1) \vee \dots \vee \neg P_{S_n}(x_n) \vee P_{S_{n+1}}(f(x_1, \dots, x_n))$ for every $f \in F$ and every $(S_1, \dots, S_n) \in SO(f)$.

The relativization C_{rel} of a clause C is the clause itself, where the sorts of the variables are ignored and for every variable x of C a literal $\neg P_{[x]}(x)$ is added to the clause. We have $CS_{rel} = A^\Sigma \cup \{C_{rel} \mid C \in CS\}$.

The following theorem, which is proved in [Sch85a], and which is proved for other signatures in [Ob62] and [Wa83], states the equivalence of the sorted and the unsorted version of a clause set:

Theorem 6. (Sortensatz) Let CS be a SIG -sorted clause set. Then:

CS is unsatisfiable iff CS_{rel} is unsatisfiable. $\square$

Example. This example is taken from [SM78], which appears to be a goldmine for theorem proving examples. During a course on automated theorem proving in the last semester, our students had to translate these puzzles into first order predicate logic and to solve them with our theorem prover (Markgraf Karl Refutation Procedure) [KM84]. One of these problems (Problem 47) reads as follows:

"When Alice entered the forest of forgetfulness, she did not forget everything, only certain things. She often forgot her name, and the most likely to forget was the day of the week. Now, the lion and the unicorn were frequent visitors to this forest. These two are strange creatures. The lion lies on Mondays, Tuesdays and Wednesdays and tells the truth on the other days of the week. The unicorn, on the other hand lies on Thursdays, Fridays and Saturdays, but tells the truth on the other days of the week.

One day Alice met the lion and the unicorn resting under a tree. They made the following statements:

Lion: Yesterday was one of my lying days.

Unicorn: Yesterday was one of my lying days.

From these statements, Alice who was a bright girl, was able to deduce the day of the week. What was it?"

We use the predicates $MO(x)$, $TU(x)$, ..., $SO(x)$ for saying that x is a Monday, Tuesday etc. Furthermore we need the binary predicate $MEMB$, indicating set Membership and a 3-ary predicate LA . $LA(x, y, z)$ is true if x says at day y that he lies at day z ; $LAYS(x)$ denotes the set of lying days of x . The remaining symbols are self explaining. One-character symbols like u, x, y, z are regarded as universally quantified variables.

Axiomatization of the days of the week:

$MO(x) \Leftrightarrow \neg(TU(x) \vee WE(x) \vee TH(x) \vee FR(x) \vee SA(x) \vee SU(x))$
 $TU(x) \Leftrightarrow \neg(WE(x) \vee TH(x) \vee FR(x) \vee SA(x) \vee SU(x) \vee MO(x))$
 $WE(x) \Leftrightarrow \neg(TH(x) \vee FR(x) \vee SA(x) \vee SU(x) \vee MO(x) \vee TU(x))$
 $TH(x) \Leftrightarrow \neg(FR(x) \vee SA(x) \vee SU(x) \vee MO(x) \vee TU(x) \vee WE(x))$
 $FR(x) \Leftrightarrow \neg(SA(x) \vee SU(x) \vee MO(x) \vee TU(x) \vee WE(x) \vee TH(x))$
 $SA(x) \Leftrightarrow \neg(SU(x) \vee MO(x) \vee TU(x) \vee WE(x) \vee TH(x) \vee FR(x))$
 $SU(x) \Leftrightarrow \neg(MO(x) \vee TU(x) \vee WE(x) \vee TH(x) \vee FR(x) \vee SA(x))$

Axiomatization of the function yesterday:

$MO(\text{yesterday}(x)) \Leftrightarrow TU(x)$
 $TU(\text{yesterday}(x)) \Leftrightarrow WE(x)$

$WE(yesterday(x)) \Leftrightarrow TH(x)$
 $TH(yesterday(x)) \Leftrightarrow FR(x)$
 $FR(yesterday(x)) \Leftrightarrow SA(x)$
 $SA(yesterday(x)) \Leftrightarrow SU(x)$
 $SU(yesterday(x)) \Leftrightarrow MO(x)$

Axiomatization of the function LDAYS:

$MEMB(x LDAYS(lion)) \Leftrightarrow MO(x) \vee TU(x) \vee WE(x)$
 $MEMB(x LDAYS(unicorn)) \Leftrightarrow TH(x) \vee FR(x) \vee SA(x)$

Axiomatization of the predicate LA:

$\neg MEMB(x LDAYS(u)) \wedge LA(u x y) \Rightarrow MEMB(y LDAYS(u))$
 $\neg MEMB(x LDAYS(u)) \wedge \neg LA(u x y) \Rightarrow$
 $\quad \neg MEMB(y LDAYS(u))$
 $MEMB(x LDAYS(u)) \wedge LA(u x y) \Rightarrow \neg MEMB(y LDAYS(u))$
 $MEMB(x LDAYS(u)) \wedge \neg LA(u x y) \Rightarrow MEMB(y LDAYS(u))$

Theorem:

$\exists x LA(lion x yesterday(x)) \wedge$
 $LA(unicorn x yesterday(x))$

The MKRP proof procedure at Kaiserslautern found a proof for this unsorted version after 183 resolution steps, among them 81 unnecessary steps, hence the final proof was 102 steps long. This proof contains a lot of trivial steps corresponding to common sense reasoning (like: if today is Monday, it is not Tuesday etc.).

Later the sort structure and the signature of the problem at hand was generated automatically by a translator module which accepts an unsorted clause set as input and produces the equivalent many-sorted version together with the corresponding signature [Sch85b].

The sort structure and the signature contain all the relevant information about the relationship of unary predicates (like our days) and the domain-rangesort relation of functions. The sort structure of the subsorts of DAYS in our example is equivalent to the lattice of subsets of {Mo, Tu, We, Th, Fr, Sa, Su} without the empty set, ordered by the subset order. Hence there are 127 ($2^7 - 1$) sorts. The function "yesterday" is a polymorphic function with 127 domain-sort relations like yesterday ((MO, WE)) = (SU, TU).

The unification algorithm exploits this information and produces only unifiers, which respect the sort relations, i.e. $(x \leftarrow t)$ is syntactically correct, if and only if the sort of the term t is less or equal the sort of the variable x . We give an example for unification:

the unifier of $x:SO \cdot TU$ and $yesterday(y:MO \cdot TU)$ is $(x \leftarrow yesterday(y_1: MO) ; y \leftarrow y_1: MO)$. The MKRP theorem-proving system [KM84] has proved the theorem in the sorted version immediately without any unnecessary steps. The length of the proof is 6. As the protocol shows, the final substitution into the theorem clause was $x \leftarrow y:Th$. Thus the ATP has found the answer, Thursday, in a very straight forward and humanlike way. Here is a proof protocol:

C1 All x:Mo MEMB (x LDAYS(lion))
 C2 All x:TU MEMB (x LDAYS(lion))
 C3 All x:We MEMB (x LDAYS(lion))
 C4 All x:Th MEMB (x LDAYS(unicorn))
 C5 All x:Fr MEMB (x LDAYS(unicorn))
 C6 All x:Sa MEMB (x LDAYS(unicorn))
 C7 All x,y:Days u:Animal
 $MEMB(x LDAYS(u)) \neg LA(u x y)$
 $MEMB(y LDAYS(u))$
 C8 All x,y:Days u:Animal
 $MEMB(x LDAYS(u)) LA(u x y)$
 $\neg MEMB(y LDAYS(u))$
 C9 All x,y:Days u:Animal
 $\neg MEMB(x LDAYS(u)) \neg LA(u x y)$
 $\neg MEMB(y LDAYS(u))$
 C10 All x,y:Days u:Animal
 $\neg MEMB(x LDAYS(u)) LA(u x y)$
 $MEMB(y LDAYS(u))$
 C11 All x:Th+Fr+Sa+Su $\neg MEMB(x LDAYS(lion))$
 C12 All x:TU+We+Su+Mo $\neg MEMB(x LDAYS(unicorn))$
 Th All x:Days
 $\neg LA(lion x yesterday(x))$
 $\neg LA(unicorn x yesterday(x))$

Proof:

C4,1 & C10,1 $\rightarrow$ R1:
 All x:Th y:Days $LA(unicorn x y)$
 $MEMB(y LDAYS(unicorn))$
 R1,2 & C12,1 $\rightarrow$ R2:
 All x:Th y:TU+We+Su+Mo $LA(unicorn x y)$
 C3,1 & C8,3 $\rightarrow$ R3:
 All x:Days y:We $MEMB(x LDAYS(lion))$
 $LA(lion x y)$
 R3,1 & C11,1 $\rightarrow$ R4:
 All x:Th+Fr+Sa+Su y:We $LA(lion x y)$
 R4,1 & Th,1 $\rightarrow$ R5:
 All x:Th $\neg LA(unicorn x yesterday(x))$
 R5,1 & R2,1 $\rightarrow$ R6: $\square$

7. Conclusion. In a sense a polymorphic calculus is more expressive than a monomorphic calculus: one function symbol essentially denotes several operations. An alternative formulation with additional equalities and different symbols surely produces a larger search space.

The ERP⁺-calculus allows to express this information and gives the possibility to use this information directly in the inference mechanism. Thus the ERP⁺-calculus has all the advantages of a many sorted calculus, but an increased expressive power.

Polymorphic unification is implemented in the MKRP theorem proving system at Kaiserslautern [KM84] and shows remarkable improvements in particular in combination with a sort generating algorithm [Sch85b], which automatically transforms a given problem into its polymorphic, many-sorted version. The ERP⁺-calculus could advantageously be used as the basis of a PROLOG implementation, where the present (Robinson) unification algorithm would be replaced by an algorithm as proposed in this paper.

Acknowledgements.

I would like to thank: my colleagues H.J. Ohlbach, A. Herold and H.J. Burckert for their support and for their helpful discussions during the preparation of this paper; Ch. Walther, who read a draft of this paper.; Prof. Ph.D. J. Siekmann, who encouraged me to write this paper and read all drafts of this paper.

References

- CD83 Cunningham, R.J., Dick, A.J.J.,
Rewrite Systems on a Lattice of Types. Rep.
No. DOC 83/7, Imperial College, London SW7
(1983)
- Co83 Cohn, A.G.
Improving the Expressiveness of Many-
sorted Logic. AAAI-83, Washington (1983)
- GM84 Goguen, J.A., Meseguer, J.
Equality, Types, Modules and Generics for
Logic Programming. Journal of Logic
Programming, (1984)
- GM85 Goguen, J.A., Meseguer, J.
Order Sorted Algebra I. Partial and
Overloaded Operators, Errors and
Inheritance. SRI Report (1985)
- Hay71 Hayes, P.
A Logic of Actions. Machine Intelligence 6,
Metamathematics Unit, University of
Edinburgh (1971)
- Hen72 Henschen, L.J.
N-Sorted Logic for Automated Theorem
Proving in Higher-Order Logic. Proc. ACM
Conference, Boston (1972)
- HO80 Huet, G., Oppen, D.C.,
Equations and Rewrite Rules, SRI Technical
Report CSL-111, (1980)
- KM84 Karl Mark G. Raph,
The Markgraf Karl Refutation Procedure,
Memo-SEKI-MK-84-01, (1984)
- Mi84 Mishra, P.
Towards a Theory of types in PROLOG. Int.
Symp. on Logic Programming (1984)
- MO84 Mycroft, A., O'Keefe, R.
A Polymorphic Type System for PROLOG.
Artificial Intelligence 23 (1984)
- Ob62 Oberschelp, A.
Untersuchungen zur mehrsortigen
Quantorenlogik. Mathematische Annalen 145
(1962)
- Ro65 Robinson, J.A. A Machine-Oriented Logic
Based on the Resolution Principle. JACM 12
(1965)
- Sch85a Schmidt-Schauss, M.
A Many-Sorted Calculus with Polymorphic
Functions Based on Resolution and
Paramodulation. Interner Bericht. Institut für
Informatik, Kaiserslautern (forthcoming)
- Sch85b Schmidt-Schauss, M.
Mechanical Generation of Sorts in Clause Sets.
Interner Bericht. Institut für Informatik,
Kaiserslautern (forthcoming)
- Sm78 Smullyan, R.M.
What is the Name of this Book? Prentice Hall
(1978)
- Wa83 Walther, C.
A Many-Sorted calculus Based on Resolution
and Paramodulation. Proc. of the 8th IJCAI,
Karlsruhe, (1983)
- Wa84 Walther, C.
Unification in Many-Sorted Theories. Proc. of
the 6th IJCAI, PISA, (1984)
- WR73 Wos, L. Robinson, G.
Maximal Models and Refutation
Completeness:
Semidecision Procedures in Automatic
Theorem Proving. In "Wordproblems"
(W.W. Boone, F.B. Cannonito, R.C. Lyndon,
eds.), North-Holland (1973)