Tableaux for dynamic hierarchies of preorders

Pedro J. Martín

Dep. de Sistemas Informáticos y Programación. Universidad Complutense de Madrid
E-mail: pjmartin@sip.ucm.es

Abstract. We present a logic for reasoning with preorders and dynamic hierarchies, where operations behave monotonically in all of their arguments. We describe and analyze two sound and complete inference systems based on semantic tableaux.

1 Introduction

The study of efficient methods for dealing with equality has been traditionally considered an important workline in different areas of theoretical computer science. Nevertheless it has recently emerged the need of extending this study to transitive relations different from those of equivalence, and of equality, in particular. This has been the case, for instance, in *CLP* [9] where constraint solvers are incorporated to logic programming. In the field of automated deduction, this situation has resulted in the development of provers with additional specific rules expressing the properties of the relation that has been studied. This line was followed for example in [2, 8, 1, 10], the two latter using resolution as refutational proof method.

On the other hand, sorts are commonly argued as a great applied benefit to approach programming closer to real world. Specifically, ordered sorts are considered useful for incorporating in a natural and elegant way, partial functions, multiple representation and constructors/selectors in structured data [7]. In automated deduction, the use of sorts entails as well a significant reduction of the search space.

When reasoning in a hierarchy of sorts, it is usual to maintain the sort information apart from data; in this sense we say that information is statically declared. However in some situations we need to reason under assumptions about sort relations, using what is known as dynamic sort information [13].

In this paper, a logic specially well suited for reasoning with preorders (reflexive and transitive relations) and dynamic hierarchies (sort relations) is defined. Operations for this logic (including predicates as boolean functions) are assumed to behave monotonically in their arguments. For this logic we present two sound and complete tableau–based deduction systems, exposed as ground and free-variable versions. Reasoning methods based on tableaux have gained attention in the past decade due mainly to theoretical and implementational progress which have permitted to build tableau–based theorem provers that can compete with those resolution–based ones. Indeed, tableaux methods can be extended to many non classical logics used in AI research, they easily allow the introduction of heuristics and human interaction, and they do not require conversion to canonical forms.

2 The logical system *LPDS*

In this section we present the logical system *LPDS* which stands for *L*ogic with *P*reorders and *D*ynamic *S*orts. Basically *LPDS* is an extended order-sorted predicate logic with the following particularities. First, data range over preordered domains and so, they can be related by using inequalities; second, sorts are dynamically ordered since the subsort information is incorporated within the language; and third, the operations are assumed to behave monotonically in each of their arguments.

Let us fix some concepts, a *preorder* is a pair $\langle D, \sqsubseteq_D \rangle$, where D is a nonempty set and $\sqsubseteq_D$ is a reflexive and transitive binary relation on D. Note that, in particular, partial orders are antisymmetric preorders, and equivalence relations (e.g. equality) are symmetric preorders. Given $(D_i, \sqsubseteq_{D_i})$, $1 \leq i \leq n$, and $(D, \sqsubseteq_D)$ preorders, a mapping $f : D_1 \times \ldots \times D_n \to D$ is *monotonic* in the i-th argument, if $f(d_1, \ldots, d_i, \ldots, d_n) \sqsubseteq_D f(d_1, \ldots, d_i', \ldots, d_n)$ for every $d_i, d_i' \in D_i$ such that $d_i \sqsubseteq_{D_i} d_i'$. The notion of monotonicity can be extended to predicates by considering that the set of boolean values $\{\underline{t}, \underline{f}\}$ is ordered by $\underline{f} \sqsubseteq \underline{t}$.

A signature Σ for *LPDS* consists of a finite set S of sorts s, and sorted sets of constants $\mathcal{C}^s$, function $\mathcal{F}^{s_1,\cdots,s_l \to s}$ and predicate symbols $\mathcal{P}^{s_1,\cdots,s_r}$. Given a signature Σ and a sorted family of variables $(X^s)_{s \in S}$, the sets of terms $T(\Sigma)$ and formulas $F(\Sigma)$ are respectively defined as follows:

$$t ::= x^s (\in X^s) \mid c^s \ (\in \mathcal{C}^s) \mid f(t_1, \ldots, t_n) \ (f \in \mathcal{F}^{s_1,\cdots,s_n \to s};\ t_i \in T(\Sigma))$$
$$\varphi ::= t \sqsubseteq t' \mid s \sqsubseteq_@ s' \mid P(t_1, \ldots, t_n) \ (P \in \mathcal{P}^{s_1,\cdots,s_n}; t_i \in T(\Sigma)) \mid \neg\varphi \mid \varphi \wedge \varphi' \mid \exists x^s \varphi$$

The formulas $\forall x^s \varphi$, $\varphi \vee \psi$ and $\varphi \to \psi$ will stand for their classical abbreviations. In *LPDS*, literals are atoms $t \sqsubseteq t'$, $s \sqsubseteq_@ s'$, $P(t_1, \ldots, t_n)$ and their negations. Ground terms and sentences are Σ-expressions where no variable occurs free. Positions, subterms and replacements in terms are defined as usual, that is $t|_p$ denotes the subterm of t at position p, and $t[s]_p$ is the result of substituting s for $t|_p$ in t.

Note that we are allowing the construction of terms and predicates which are not well-sorted w.r.t. the signature. The reason for this apparent misuse is to allow *LPDS* to express properties about data, that are conditioned to assumptions about sorts. For example, the formula $s \sqsubseteq_@ s' \to \forall x^s \exists x^{s'} (x^s \sqsubseteq x^{s'})$ will be interpreted as true since the consequent of the implication can be derived from the truth of the antecedent. When we make reasonings, the combination of the subsort information provided by the formulas and the sort information given in the signature drastically reduces the search space.

Given a set Φ of subsort relations of the form $s \sqsubseteq_@ s'$, what we call a (dynamic) *hierarchy*, we can define whether a term t is *well-sorted* w.r.t. Φ or not. For example, if $f \in F^{s_1 \to s_2}, a \in C^s$ is given in the signature, then $f(a)$ is a well-sorted term w.r.t. $\{s \sqsubseteq_@ s', s' \sqsubseteq_@ s_1\}$.

Definition 1. *Given a hierarchy Φ, the sorted family of Σ-terms that are well-sorted w.r.t. Φ, written $(\mathcal{T}_\Sigma^\Phi(s))_{s \in S}$, is the least sorted family of sets $(Y^s)_{s \in S}$ of Σ-terms such that:*

1. $\mathcal{C}^s \subset Y^s$ *2.* $X^s \subset Y^s$ *3. If* $s' \sqsubseteq_@ s \in \Phi$ *then* $Y^{s'} \subseteq Y^s$
4. If $f \in \mathcal{F}^{s_1,\cdots,s_n \to s}$ *and* $t_i \in Y^{s_i}$, $1 \leq i \leq n$, *then* $f(t_1, \ldots, t_n) \in Y^s$

If $t \in \mathcal{T}_\Sigma^\Phi(s)$ then we will say that t has *dynamic* sort s, because the sort of t depends on the subsort information of Φ. On the contrary, we use $sort(t)$ for representing the sort of a term t that is deduced from the (static) sort information contained in the signature. It is defined by $sort(c^s) = sort(x^s) = s$, and $sort(f(t_1, \ldots, t_n)) = s$, if $f \in \mathcal{F}^{s_1,\cdots,s_n \to s}$. If $sort(t) = s$, we will say that t has *static* sort s. In the example above, a has static sort s and dynamic sorts s, s' and s_1 w.r.t. $\{s \sqsubseteq_@ s', s' \sqsubseteq_@ s_1\}$.

Given a hierarchy Φ, we say that the sort s is a *subsort* of s' *w.r.t.* Φ, written $s \ll^\Phi s'$, if either $s' = s$ or there exists a sequence of formulas in Φ of the form $s \sqsubseteq_@ s_1, s_1 \sqsubseteq_@ s_2, \ldots, s_k \sqsubseteq_@ s'$. The relation $\ll^\Phi$ is decidable for finite hierarchies Φ and can be computed in $O(size(S)^3)$. Dynamic and static sorts are related as follows.

Lemma 1. *1.* $t \in \mathcal{T}_\Sigma^\Phi(s) \iff (sort(t) \ll^\Phi s$ *and* $t \in \mathcal{T}_\Sigma^\Phi(sort(t)))$.
2. If $f \in F^{s_1,\cdots,s_n \to s}$ *and* $t = f(t_1, \ldots, t_n) \in \mathcal{T}_\Sigma^\Phi(s)$ *then* $t_i \in \mathcal{T}_\Sigma^\Phi(s_i), 1 \leq i \leq n$.

This result has an important consequence from a practical point of view: the problem "$t \in \mathcal{T}_\Sigma^\Phi(s)$?" is decidable, assuming the finiteness of Φ. In fact such a problem can be solved in polynomial time w.r.t. the sizes of t and S. For example, if $f \in F^{s_1 \to s_2}$, then we can prove that $f(t') \in \mathcal{T}_\Sigma^\Phi(s)$, by checking whether $s_2 \ll^\Phi s$ –which is decidable for a finite Φ– and proving that $t' \in \mathcal{T}_\Sigma^\Phi(s_1)$. The notion of *well-sortedness* can be naturally extended from terms to formulas.

Definition 2. *The well-sortedness of a formula* φ *w.r.t. a hierarchy* Φ, *written* $WS(\varphi, \Phi)$, *is defined as follows:*

1. $WS(s \sqsubseteq_@ s', \Phi) \Leftrightarrow s \sqsubseteq_@ s' \in \Phi$
2. $WS(t_1 \sqsubseteq t_2, \Phi) \Leftrightarrow$ *there exists* $s \in S$ *such that* $t_i \in \mathcal{T}_\Sigma^\Phi(s)$, $i = 1, 2$
3. $WS(P(t_1, \ldots, t_n), \Phi) \Leftrightarrow t_i \in \mathcal{T}_\Sigma^\Phi(s_i)$, $1 \leq i \leq n$, $P \in \mathcal{P}^{s_1 \ldots s_n}$
4. $WS(\neg\varphi, \Phi) \Leftrightarrow WS(\varphi, \Phi)$
5. $WS(\varphi_1 \wedge \varphi_2, \Phi) \Leftrightarrow WS(\varphi_i, \Phi), i = 1, 2$
6. $WS(\exists x^s \varphi, \Phi) \Leftrightarrow WS(\varphi, \Phi)$

Given a set of formulas Ψ, $WS(\Psi, \Phi)$ *holds if* $WS(\varphi, \Phi)$, *for every* $\varphi \in \Psi$.

Again Lemma 1 implies that the problem "$WS(\varphi, \Phi)$?" is also decidable whenever Φ is finite (and polynomial w.r.t. the sizes of φ and S).

As in first-order logic, substitutions in *LPDS* are mappings τ from variables to terms such that the domain of τ defined by $dom(\tau) = \{x^s \in X / \tau(x^s) \neq x^s\}$, is finite. A substitution with domain $\{x_1, \ldots, x_k\}$ will be represented by $[\tau(x_1)/x_1, \ldots, \tau(x_k)/x_k]$. In order to preserve the soundness of the tableau systems we present, we must require that the static sort s of the substituted variable x^s is a dynamic sort of the substituted term $\tau(x^s)$.

Definition 3. *A substitution* τ *is well-sorted w.r.t. a hierarchy* Φ, *written* $WS(\tau, \Phi)$, *if* $\tau(x^s) \in \mathcal{T}_\Sigma^\Phi(s)$.

Among the main properties of well-sorted substitutions, it can be proved that they preserve the dynamic sorts of terms and the well-sortedness of formulas.

Lemma 2. *Given a term t, a formula φ, a hierarchy Φ, and a substitution τ such that $WS(\tau, \Phi)$, the following properties hold:*

1. If $t \in \mathcal{T}_\Sigma^\Phi(s)$, then $t\tau \in \mathcal{T}_\Sigma^\Phi(s)$ *2. If $WS(\varphi, \Phi)$, then $WS(\varphi\tau, \Phi)$*

The reciprocal results also hold whenever the substitution preserves static sorts. Formally, a substitution τ *preserves static sorts* if $sort(\tau(x^s)) = s$.

Lemma 3. *Given terms t, t', a formula φ, a hierarchy Φ, and a substitution τ which preserves static sorts, the following properties hold:*

1. If $t\tau \in \mathcal{T}_\Sigma^\Phi(s)$, then $t \in \mathcal{T}_\Sigma^\Phi(s)$ *2. If $WS(\varphi\tau, \Phi)$, then $WS(\varphi, \Phi)$*

3. If $t\tau = t'\tau$, $WS(\tau, \Phi)$ and $\theta = mgu(t, t')$[1], then $WS(\theta, \Phi)$.

3 Semantics of *LPDS*

In order to interpret terms, Σ-structures in *LPDS* must supply preordered domains for each sort, and a special new element $\perp$ for representing the value of non well-sorted terms.

Definition 4. *A Σ-structure $\mathcal{D}$ is composed of a system $\{(D^s, \sqsubseteq_s^\mathcal{D}) | s \in S\} \cup \{\perp\}$ and interpretations for constants $\{c^\mathcal{D} \in D^s | c \in \mathcal{C}^s\}$, function $\{f^\mathcal{D} : D^{s_1} \times \ldots \times D^{s_l} \to D^s | f \in \mathcal{F}^{s_1,\ldots,s_l \to s}\}$ and predicate symbols $\{P^\mathcal{D} : D^{s_1} \times \ldots \times D^{s_r} \to \{\underline{f}, \underline{t}\} | P \in \mathcal{P}^{s_1,\ldots,s_r}\}$ such that:*

1. *$(D^s, \sqsubseteq_s^\mathcal{D})_{s \in S}$ is a sorted family of preorders which is* transitive, *that is: (†) if $d \sqsubseteq_s^\mathcal{D} d'$, $d' \sqsubseteq_{s'}^\mathcal{D} d''$, and $d, d'' \in D^{s''}$, then $d \sqsubseteq_{s''}^\mathcal{D} d''$.*
2. *$f^\mathcal{D}$ and $P^\mathcal{D}$ are monotonic in each of their arguments.*

A valuation ρ for $\mathcal{D}$ is a sorted family of mappings $\rho^s : X^s \to D^s$. A Σ-interpretation is $\langle \mathcal{D}, \rho \rangle$ where $\mathcal{D}$ is a Σ-structure and ρ is a valuation for $\mathcal{D}$.

We make two observations. First, the valuation of variables only range over non-$\perp$ values. Second, we require transitivity (†) in the family of preorders in order to ensure the soundness of some tableau rules (cfr. ζ later on); this avoids some pathological structures, allowing to prove, for example, that for every sorts s, s', if $d, d' \in D^s \cap D^{s'}$ and $d \sqsubseteq_s^\mathcal{D} d'$, then $d \sqsubseteq_{s'}^\mathcal{D} d'$.

Definition 5. *Let $\langle \mathcal{D}, \rho \rangle$ be a Σ-interpretation. The semantic value of a Σ-term t, with $sort(t) = s$, in $\langle \mathcal{D}, \rho \rangle$ is an element $[\![t]\!]_\rho^\mathcal{D} \in D^s \cup \{\perp\}$ defined by: (1) $[\![c^s]\!]_\rho^\mathcal{D} = c^\mathcal{D}$; (2) $[\![x^s]\!]_\rho^\mathcal{D} = \rho(x)$, and (3) for every $f \in \mathcal{F}^{s_1,\ldots,s_n \to s}$:*

$$[\![f(t_1, \ldots, t_n)]\!]_\rho^\mathcal{D} = \begin{cases} f^\mathcal{D}([\![t_1]\!]_\rho^\mathcal{D}, \ldots, [\![t_n]\!]_\rho^\mathcal{D}) & \text{if } [\![t_i]\!]_\rho^\mathcal{D} \in D^{s_i}, 1 \le i \le n \\ \perp & \text{otherwise.} \end{cases}$$

The value of a Σ-formula φ in $\langle \mathcal{D}, \rho \rangle$, written $[\![\varphi]\!]_\rho^\mathcal{D} (\in \{\underline{f}, \underline{t}\})$, is defined by:

- $[\![s \sqsubseteq_@ s']\!]_\rho^\mathcal{D} = \begin{cases} \underline{t} & \text{if } D^s \subseteq D^{s'} \\ \underline{f} & \text{otherwise} \end{cases}$

[1] $mgu(t, t')$ is the most general unifier of t and t'.

- $[\![t_1 \sqsubseteq t_2]\!]^{\mathcal{D}}_{\rho} = \begin{cases} \underline{t} & \text{if there is } s \text{ such that } ([\![t_i]\!]^{\mathcal{D}}_{\rho} \in D^s, i = 1,2, \text{ and } [\![t_1]\!]^{\mathcal{D}}_{\rho} \sqsubseteq^{\mathcal{D}}_{s} [\![t_2]\!]^{\mathcal{D}}_{\rho}) \\ \underline{f} & \text{otherwise} \end{cases}$
- $[\![P(t_1, \ldots, t_n)]\!]^{\mathcal{D}}_{\rho} = \begin{cases} P^{\mathcal{D}}([\![t_1]\!]^{\mathcal{D}}_{\rho}, \ldots, [\![t_n]\!]^{\mathcal{D}}_{\rho}) & \text{if } [\![t_i]\!]^{\mathcal{D}}_{\rho} \in \mathcal{D}^{s_i}, 1 \leq i \leq n \\ \underline{f} & \text{otherwise} \end{cases}$
  for every $P \in \mathcal{P}^{s_1,\ldots,s_n}$.
- *The boolean value for* $\neg, \wedge$ *and* $\exists$ *is defined as in first-order logic.*

The concepts of model, satisfiability and logical consequence can be defined as in first-order logic. Finally, the following lemma presents a collection of results that will be important when proving the properties of the tableau systems of the next sections.

Lemma 4. *1. Given a hierarchy* Φ *and a term* t*, if* $\langle \mathcal{D}, \rho \rangle \models \Phi$ *and* $t \in \mathcal{T}^{\Phi}_{\Sigma}(s)$*, then* $[\![t]\!]^{\mathcal{D}}_{\rho} \in D^s$.

2. Given a hierarchy Φ *and a substitution* $\tau = [t_1/x_1, \ldots, t_n/x_n]$*, if* $\langle \mathcal{D}, \rho \rangle \models \Phi$ *and* $WS(\tau, \Phi)$*, then the following holds: (1)* $[\![t\tau]\!]^{\mathcal{D}}_{\rho} = [\![t]\!]_{\rho'}$*; and (2)* $[\![\varphi\tau]\!]^{\mathcal{D}}_{\rho} = [\![\varphi]\!]_{\rho'}$*, where* $\rho' = \rho[[\![t_1]\!]^{\mathcal{D}}_{\rho}/x_1, \ldots, [\![t_n]\!]^{\mathcal{D}}_{\rho}/x_n]$.

3. If $\langle \mathcal{D}, \rho \rangle \models t_1 \sqsubseteq t_2$ *and exists* s *such that* $[\![t[t_i]_p]\!]^{\mathcal{D}}_{\rho} \in D^s$*,* $i = 1,2$*, then* $[\![t[t_1]_p]\!]^{\mathcal{D}}_{\rho} \sqsubseteq^{\mathcal{D}}_{s} [\![t[t_2]_p]\!]^{\mathcal{D}}_{\rho}$.

4. If $\langle \mathcal{D}, \rho \rangle \models t^1 \sqsubseteq t^2$*,* $[\![t_i[t^j]_p]\!]^{\mathcal{D}}_{\rho} \in D^{s_i}$*,* $j = 1,2$*, and* $[\![P(t_1, \ldots, t_i[t^1]_p, \ldots, t_n)]\!]^{\mathcal{D}}_{\rho}$*, then* $[\![P(t_1, \ldots, t_i[t^2]_p, \ldots, t_n)]\!]^{\mathcal{D}}_{\rho}$.

4 Ground tableaux

In order to deal with the preorder relation and the sort information, we propose inference systems that extend classical tableau methods. As usual, a tableau for a finite set of *LPDS*-sentences Φ is a formula-labeled tree which, begining with a single branch with a node for every $\varphi \in \Phi$, grows up by the application of rules to the formulas of its branches. The ground tableau method we present includes the classical α and β rules [5], respectively used to extend (α applied to $\varphi \wedge \psi$ or $\neg\neg\varphi$) or split (β applied to $\neg(\varphi \wedge \psi)$) a branch, and incorporate the rules of Table 1 for preorders and sorts. Specifically, we introduce the rules *Ref* and ζ for the preorder relation and the monotonic behavior of function sysmbols, and ξ for the monotonicity of predicate sysmbols. For sorts, we adapt the usual rules γ and δ involved in the expansion of quantified formulas. In δ, we suppose that the signature is extended to contain collections of infinite sets AC^s and $SF^{s_1,\ldots,s_l \to s}$, $s_1, \ldots, s_l, s \in S$, of auxiliary constants and Skolem function symbols, respectively. Note that, for simplicity, we use the branch B instead of the hierarchy included in B.

Observe that only "well-sorted" information is added to a tableau when the rules of Table 1 are applied. This trivially holds for the rules *Ref*, ζ and ξ since such a condition is explicitely required. It is also true for the rules γ and δ because the introduced terms are well-sorted w.r.t. the related branch.

A common characteristic of tableau systems is that they are refutational methods; that is, they study whether a set of sentences Φ is satisfiable or not.

(γ)	If $\neg\exists x^s\varphi \in B$, then B is enlarged with a new node labeled by $\neg\varphi[t/x^s]$, where t is a ground term such that $t \in \mathcal{T}_\Sigma^B(s)$
(δ)	If $\exists x^s\varphi \in B$, then B is enlarged with a new node labeled by $\varphi[t/x^s]$, where t is a ground term such that $t \in \mathcal{T}_\Sigma^B(s)$ and its root symbol is new to B
(*Ref*)	B is enlarged with the new node labeled by $t \sqsubseteq t$, where t is a ground term such that $WS(t \sqsubseteq t, B)$
(ζ)	If $t \sqsubseteq t'$, $t_1 \sqsubseteq t_2[t]_p \in B$, then B is enlarged with a new node labeled by $t_1 \sqsubseteq t_2[t']_p$, whenever $WS(t_1 \sqsubseteq t_2[t']_p, B)$
(ξ)	If $P(t_1, \ldots, t_i[t]_p, \ldots, t_n)$, $t \sqsubseteq t' \in B$, then B is enlarged with a new node labeled by $P(t_1, \ldots, t_i[t']_p, \ldots, t_n)$, whenever $WS(P(t_1, \ldots, t_i[t']_p, \ldots, t_n), B)$

Table 1. Rules of $\mathcal{G}$

The branches of a tableau for Φ represent all the available cases corresponding to the hypothesis "Φ is satisfiable". Therefore Φ is not satisfiable if none of these branches is satisfiable. To syntactically prove such a condition, we introduce a usual *closure rule*: *a branch B is closed, and then it is not extended and not splitted anymore, if two literals of the form φ and $\neg\varphi$ occur in B.* We also say that a branch is open if it is not closed, and that a tableau is closed whenever all of its branches are closed.

Let $\mathcal{G}$ (for *Ground*) be the tableau system composed of the rules α, β, *closure*, and the rules of Table 1. We say that it is ground because the application of the rule γ replaces the universally quantified variable x^s by a ground term with the same dynamic sort w.r.t. the related branch.

The main properties of $\mathcal{G}$ are its soundness and its completeness. It is sound because a set of sentences Φ is not satisfiable whenever a closed $\mathcal{G}$-tableau can be built for Φ. As for the case of first-order logic, this property is based on the fact that the satisfiability of a tableau $\mathcal{T}$, that is the existence of a satisfiable branch, remains when an expansion rule is applied to $\mathcal{T}$.

Theorem 1 (Soundness of $\mathcal{G}$). *Let Φ be a finite set of Σ-sentences. If there exists a closed $\mathcal{G}$-tableau for Φ, then Φ is not satisfiable.*

The opposite direction in the previous theorem corresponds to the completeness of the system, but for *LPDS* we need an additional hypothesis. If there is no closed $\mathcal{G}$-tableau for the set Φ, it is obvious that we can systematically build a tableau for Φ which expands all of its open branches at length. Then, it can be proved that if some branch B is still open, B has enough information to define a model of $\{\varphi \in B / WS(\varphi, B)\}$. In order to ensure that the initial set of sentences Φ is well-sorted w.r.t. any of these branches, Φ is required to be well-sorted w.r.t. the hierarchy statically obtained from itself.

Definition 6. *The static hierarchy of a sentence φ, written $\mathcal{P}(\varphi)$, is recursively defined by: (1) $\mathcal{P}(s \sqsubseteq_@ s') = \{s \sqsubseteq_@ s'\}$, (2) $\mathcal{P}(\varphi) = \emptyset$, for another literal φ, (3) $\mathcal{P}(\varphi \wedge \psi) = \mathcal{P}(\varphi) \cup \mathcal{P}(\psi)$, (4) $\mathcal{P}(\neg(\varphi \wedge \psi)) = \emptyset$, (5) $\mathcal{P}(\exists x\varphi) = \mathcal{P}(\varphi)$, (6) $\mathcal{P}(\neg\exists x\varphi) = \mathcal{P}(\neg\varphi)$ and (7) $\mathcal{P}(\neg\neg\varphi) = \mathcal{P}(\varphi)$. For a set of sentences Φ, $\mathcal{P}(\Phi) = \bigcup_{\varphi\in\Phi} \mathcal{P}(\varphi)$.*

Theorem 2 (Completeness of $\mathcal{G}$). *If Φ is a finite set of Σ-sentences not satisfiable and $WS(\Phi, \mathcal{P}(\Phi))$, then a closed $\mathcal{G}$-tableau for Φ can be built.*

Soundness and completeness hold even when extra conditions are required in rule δ. For example, it is frequent to replace the existentially quantified variable x^s by a new constant $c^s \in AC^s$. However, in order to allow the tableau system we present in the next section to simulate $\mathcal{G}$, it is more convenient to introduce the ground term $f(t_1, \ldots, t_n)$, where $f \in SF^{s_1,\ldots,s_n \to s}$ is new to the branch B, $t_1, \ldots, t_n$ are the terms previously introduced to B by a γ-application and $s_i = sort(t_i)$, $1 \leq i \leq n$.

5 Free-variable tableaux

Analyzing the system $\mathcal{G}$ one finds that one of its main drawbacks comes up when guessing the ground term introduced in the application of the rule γ. It is better to use a free-variable to represent a unique but unknown datum. The specific value of this variable will be decided in the unification process involved when closing a branch or when applying the rules related to the preorder relation. Observe that its instantiation must be coherent with the sort information, that is, a free-variable x^s has to be substituted by a term $t \in \mathcal{T}_\Sigma^B(s)$, for every branch B where x^s occurs. Only substitutions satisfying such a requirement are allowed for *LPDS* in order to preserve soundness.

Definition 7. *A substitution σ is well-sorted w.r.t. a tableau $\mathcal{T}$, written $WS(\sigma, \mathcal{T})$, if $WS(\sigma|_{var(B)}, B)$, for every branch B of $\mathcal{T}$.*

Let $\mathcal{FV}$ (for *Free-Variable*) be the tableau system composed of the rules α, β and the rules of Table 2. In order to close a tableau, we compute a most general unifier and test if it is well-sorted w.r.t. the whole tableau. Hence $\mathcal{FV}$ also includes the following *closure rule*: *a tableau $\mathcal{T}$, composed of branches $B_1, \ldots, B_k$, is closed if there exists a set of equations of atomic formulas $M = \{\varphi_1 \simeq \psi_1, \ldots, \varphi_k \simeq \psi_k\}$ such that M is unifiable, $WS(mgu(M), \mathcal{T})$ and $\varphi_i, \neg\psi_i \in B_i, 1 \leq i \leq k$.*

Observe that, given a substitution θ and a tableau $\mathcal{T}$, the problem "$WS(\theta, \mathcal{T})$?" is decidable because the question "$t \in \mathcal{T}_\Sigma^B(s)$?" can be solved in polynomial time for every branch B of $\mathcal{T}$. Hence the requirements needed for the application of the closure rule once M is chosen, that is, the computation of $mgu(M)$ and its well-sortedness test, can be solved polynomially.

The system $\mathcal{FV}$ is sound and complete. For the case of soundness, Theorem 1 can be adapted since the existence of a model for a $\mathcal{FV}$-tableau is again preserved when it is extended. However the notion of satisfiability for tableaux must be redefined to deal with free-variables. Briefly, we use the notion of *ground satisfiability*: there exists a Σ-structure which models $\mathcal{T}\tau$ for every ground substitution τ such that $var(\mathcal{T}) \subseteq dom(\tau)$ and $WS(\tau, \mathcal{T})$.

Theorem 3 (Soundness of $\mathcal{FV}$). *If a finite set of Σ-sentences Φ has a closed $\mathcal{FV}$-tableau, then Φ is not satisfiable.*

(γ') If $\neg\exists x^s\varphi \in B$ and $s' \ll^B s$, then B is enlarged with a new node labeled with $\neg\varphi[y^{s'}/x^s]$, where $y^{s'}$ is a new free-variable to the tableau

(δ') If $\exists x^s\varphi \in B$, then B is enlarged with a new node labeled with $\varphi[f(x_1^{s_1},\ldots,x_n^{s_n})/x^s]$, where $f \in SF^{s_1,\ldots,s_n\to s}$ is new to B and $x_1^{s_1},\ldots,x_n^{s_n}$ are the free-variables of B

(ζ') If $t \sqsubseteq t'$, $t_1 \sqsubseteq t_2[t'']_p \in B$, $\sigma = mgu(t,t'')$ and $WS(\sigma,\mathcal{T})$, then σ is applied to $\mathcal{T}$ and $B\sigma$ is enlarged with a new node labeled with $(t_1 \sqsubseteq t_2[t']_p)\sigma$, whenever $WS((t_1 \sqsubseteq t_2[t']_p)\sigma, B)$

(ξ') If $t \sqsubseteq t'$, $P(t_1,\ldots,t_i[t'']_p,\ldots,t_n) \in B$, $\sigma = mgu(t,t'')$ and $WS(\sigma,\mathcal{T})$, then σ is applied to $\mathcal{T}$ and $B\sigma$ is enlarged with a new node labeled with $P(t_1,\ldots,t_i[t']_p,\ldots,t_n)\sigma$, whenever $WS(P(t_1,\ldots,t_i[t']_p,\ldots,t_n)\sigma, B)$

(*Refv*) B is enlarged with a new node labeled with $x^s \sqsubseteq x^s$, where x^s is new to $\mathcal{T}$

(*Reff*) B is enlarged with a new node labeled with $f(x_1^{s_1},\ldots,x_n^{s_n}) \sqsubseteq f(x_1^{s_1},\ldots,x_n^{s_n})$, where f is a function symbol, $x_1^{s_1},\ldots,x_n^{s_n}$ are new free-variables to $\mathcal{T}$ and $WS(f(x_1^{s_1},\ldots,x_n^{s_n}) \sqsubseteq f(x_1^{s_1},\ldots,x_n^{s_n}), B)$

Table 2. Rules of $\mathcal{FV}$

$\mathcal{FV}$ completeness can be proved by lifting a closed $\mathcal{G}$-tableau, which is provided by Theorem 2. Before presenting the corresponding theorem, we prove that the application of the $\mathcal{G}$-rules *Ref*, ζ and ξ can be simulated one by one within the system $\mathcal{FV}$.

Lemma 5 (Lifting). *Let $\mathcal{T}$ and τ be a $\mathcal{FV}$-tableau and a ground substitution such that $var(\mathcal{T}) \subseteq dom(\tau)$, $WS(\tau,\mathcal{T})$ and τ preserves static sorts. If $\mathcal{T}\tau$ can be closed using the rules Ref, ζ and ξ, then $\mathcal{T}$ can be closed using Refv, Reff, ζ' and ξ'.*

Theorem 4 (Completeness of $\mathcal{FV}$). *If Φ is a finite set of Σ-sentences not satisfiable and $WS(\Phi,\mathcal{P}(\Phi))$, then a closed $\mathcal{FV}$-tableau for Φ can be built.*

Proof. By Theorem 2 there exists a closed $\mathcal{G}$-tableau $\mathcal{T}_g$ for Φ where any δ-expansion introduces a functional term as explained at the end of Section 4. We transform $\mathcal{T}_g$ into a new closed $\mathcal{G}$-tableau $\mathcal{T}_g'$, by moving the applications of the rules *Ref*, ζ and ξ to the end of every branch where they occur. Note that this can be done because (a) only literals are involved in these rules, hence we can move up the application of the rules $\alpha,\beta,\gamma,\delta$; ($b$) we can apply repeatedly the rules *Ref*, ζ and ξ when rising a β-expansion in order to force them to occur in every branch produced by the disjunction.

Let $\mathcal{T}$ be the $\mathcal{FV}$-tableau which simulates each γ-application to $\mathcal{T}_g'$ that introduces the ground term $t \in \mathcal{T}_\Sigma^B(s)$, by applying γ' to $\mathcal{T}$ to introduce the free-variable $x^{sort(t)}$. Let σ be the substitution that groundly instances these free-variables with the corresponding ground term of $\mathcal{T}_g'$, that is, $\sigma(x^s) = t$. Hence σ relates both tableaux because $\mathcal{T}\sigma = \mathcal{T}_g'$. Moreover, $WS(\sigma,\mathcal{T})$ holds and σ preserves static sorts. Thus, Lemma 5 can be used to conclude that $\mathcal{T}$ can be closed. ■

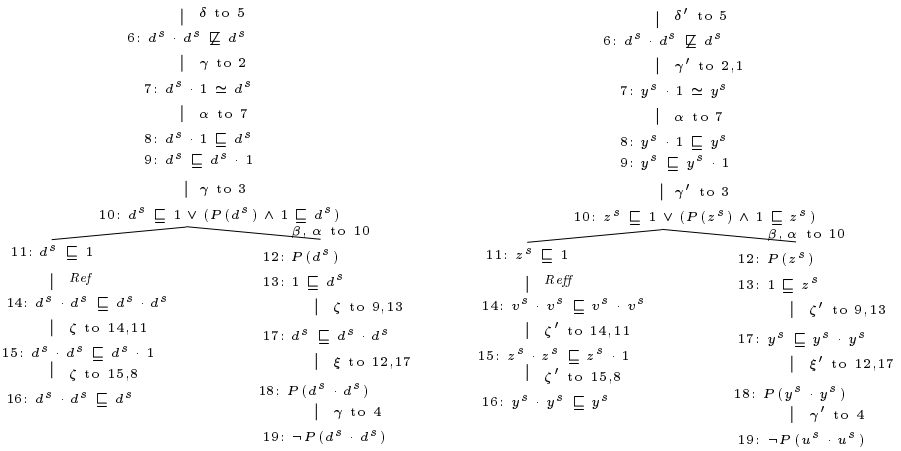

Fig. 1. The $\mathcal{G}$-tableau $\mathcal{T}_g$ (left) and the $\mathcal{FV}$-tableau $\mathcal{T}$ (right)

Example 1. Given two sorts s and $\mathbb{R}^+$, let $1^{\mathbb{R}^+}, \cdot^{\mathbb{R}^+,\mathbb{R}^+\to\mathbb{R}^+}$ *and* $P^{\mathbb{R}^+}$ be a constant, a function and a predicate symbol, respectively. Then $\psi = \forall x^s(x^s \cdot x^s \sqsubseteq x^s)$ follows from the set of hypothesis $\Phi = \{1 : s \sqsubseteq_@ \mathbb{R}^+, 2 : \forall x^{\mathbb{R}^+}(x^{\mathbb{R}^+} \cdot 1 \simeq x^{\mathbb{R}^+}), 3 : \forall x^s(x^s \sqsubseteq 1 \vee (P(x^s) \wedge 1 \sqsubseteq x^s)), 4 : \forall x^s \neg P(x^s \cdot x^s)\}$. Note that we use the symbol $\cdot$ in infix form and $\Box \simeq \Diamond$ as shorthand for $\Box \sqsubseteq \Diamond \wedge \Diamond \sqsubseteq \Box$.

In Figure 1 on the left, we show the simplified $\mathcal{G}$-tableau $\mathcal{T}_g$ for $\Phi \cup \{5 : \neg\psi\}$, which is closed by 6, 16, 18 and 19. On the right, the closed $\mathcal{FV}$-tableau $\mathcal{T}$ simulates $\mathcal{T}_g$ (cfr. proof of Theorem 4). Note that $\mathcal{T}$ changes when the rule ζ' is applied to obtain 15 and 16, since the substitutions $[z^s/v^s]$ and $[y^s/z^s]$, which are well-sorted, are respectively applied to the whole tableau. For example, formulas 12 and 13 respectively turn into $P(y^s)$ and $1 \sqsubseteq y^s$ before introducing 17. Finally, $\mathcal{T}$ is closed because the substitution $\theta = mgu(\{16 : y^s \cdot y^s \sqsubseteq y^s \simeq 6 : d^s \cdot d^s \sqsubseteq d^s, 18 : P(y^s, y^s) \simeq 19 : P(u^s, u^s)\}) = [d^s/y^s, d^s/u^s]$ satisfies $WS(\theta, \mathcal{T})$.

6 Analysis and related works

Let us analyze the system $\mathcal{FV}$ from a practical point of view. One of its main advantages arises when dealing with sort information, since $\mathcal{FV}$ only requires standard unification and a test to ensure well-sortedness. Thus, the cost of using dynamic hierarchies is polynomial and so its introduction does not increase the complexity of classical free-variable tableaux for first-order logic. Respecting sorts, the unique drawback of the system comes from the application of the rule γ', since the universally quantified variable x^s can be replaced by a new free-variable of *any* subsort of s. Other approaches integrate sorts into first-order logic in a more expressive way [13, 12], but at a high cost: their free-variable tableaux use specific procedures of (rigid and sorted) unification which are terminating, although they do not ensure polynomial complexity.

The main potential inefficiency of the system $\mathcal{FV}$ results from the application of the monotonicity and preorder rules $(\zeta', \xi', \mathit{Refv}, \mathit{Reff})$. Obviously, heuristics are essential to control their application. Among these rules, *Refv* and *Reff* are critical since they are axioms, that is, their application does not need any information of the branch. It has been proved that such axioms can be avoided for equality [3, 4], but for monotonic preorders they are necessary (for example, to prove that the set $\{a \sqsubseteq b, \forall x^s \forall y^s f(g(b), x^s, x^s) \not\sqsubseteq f(y^s, y^s, g(a))\}$ is not satisfiable, where $a, b \in \mathcal{C}^s$). [11] describes a procedure of (rigid monotonic and preordered) unification which avoids these axioms, but it is non terminating.

[6] presents a preliminary three–valued version of the logic *LPDS*, which includes the semantic value $\underline{u}$ to interpret non well-sorted formulas. This approach introduces some changes in the tableau rules. For example, the premises of the rules ζ' and ξ' ($t \sqsubseteq t'$, $t_1 \sqsubseteq t_2[t'']_p$ and $P(t_1, \ldots, t_i[t'']_p, \ldots, t_n)$, respectively) are required to be well-sorted, instead of their conclusions. Closing a branch, [6] asks for a pair of well-sorted literals, otherwise they could be interpreted to $\underline{u}$ and no semantic contradiction would hold. Another difference corresponds to the computation of the family $(\mathcal{T}^{\Phi}_{\Sigma}(s))_{s \in S}$, because the closure of the subsort relation $\ll^{\Phi}$ is explicitly obtained in [6] by means of two rules: $\mathit{Ref}_{@}$, used to add $s \sqsubseteq_{@} s$ to any branch, and $\mathit{Tran}_{@}$ used to introduce $s_1 \sqsubseteq_{@} s_3$ in a branch B, whenever $s_1 \sqsubseteq_{@} s_2, s_2 \sqsubseteq_{@} s_3 \in B$.

Proofs of the present paper are available at *http://babel.dacya.ucm.es/chus/tr02.ps.gz*

References

1. L. Bachmair, H. Ganzinger. *Rewrite techniques for transitive relations.* Procs. LICS'94, 384–393, 1994.
2. W. W. Bledsoe, K. Kunen, R. Shostak. *Completeness Results for Inequality Provers.* Artificial Intelligence 27, 255-288, 1985.
3. D. Brand. *Proving Theorems with the Modification Method.* SIAM J. Computation 4(4), 412–430, 1975.
4. A. Degtyarev, A. Voronkov. *What you always wanted to know about rigid E-unification.* Journal of Automated Reasoning 20(1), 47–80, 1998.
5. M. Fitting. *First-Order Logic and Automated Theorem Proving.* Springer, 1996.
6. A. Gavilanes, J. Leach, P. J. Martín, S. Nieva. *Reasoning with preorders and dynamic sorts using free variable tableaux.* AISMC-3. LNCS 1138, 365–379, 1996.
7. J. A. Goguen, J. Meseguer. *Order-sorted algebra I.* Theoretical Computer Science 105, 217–273, 1992.
8. L. M. Hines. *The Central Variable Strategy of Str+ve.* CADE'11, LNAI 607, 35-49, Springer Verlag, 1992.
9. J. Jaffar, M. J. Maher. *Constraint logic programming: A survey.* Journal of Logic Programming 19/20, 503–582, 1994.
10. J. Levy, J. Agustí. *Bi-rewrite systems.* J. of Symb. Computation 22, 279–314, 1996.
11. P.J. Martín, A. Gavilanes. *Monotonic preorders for free variable tableaux.* Procs. TABLEAUX'2000, LNAI 1847, 309–323, 2000.
12. P. J. Martín, A. Gavilanes, J. Leach. *Tableau Methods for a Logic with Term Declarations.* J. of Symbolic Computation 29, 343–372, 2000.
13. C. Weidenbach. *First-order tableaux with sorts.* J. of the Interest Group in Pure and Applied Logics 3(6), 887–907, 1995.