Skolem functions and Hilbert's ϵ-terms in Free Variable Tableau Systems

Domenico Cantone and Marianna Nicolosi Asmundo

Dipartimento di Matematica e Informatica, Università di Catania
Viale A. Doria 6, I-95125 Catania, Italy
e-mail: cantone@dmi.unict.it, nicolosi@dmi.unict.it

Abstract. We apply the framework of the generic $\boldsymbol{\delta}$-rule presented in [4] to show how to map the $\boldsymbol{\delta}^{\epsilon}$-rule, which uses ϵ-terms as syntactical objects to expand existentially quantified formulae, in the context of standard $\boldsymbol{\delta}$-rules based on Skolem terms. Structural results coming out from such mapping process are discussed.

1 Introduction

Elimination of existential quantifiers in tableau proofs is dealt with by the $\boldsymbol{\delta}$-rule expansion rule, which replaces existentially quantified variables with suitable terms.

The $\boldsymbol{\delta}$-rule was introduced in [22], in the context of ground tableaux, by enriching the signature with a countable collection of new parameters of arity 0. Since then, after the introduction of free variables in semantic tableaux, the $\boldsymbol{\delta}$-rule has undergone several liberalizations aimed at increasing the efficiency of the underlying proof system. Most $\boldsymbol{\delta}$-rule variants are based on the Skolemization technique to produce the terms to be substituted for the existentially quantified variables [10, 13, 2, 1, 3].

Skolemization is one of the most widespread techniques for the elimination of existential quantifiers in the context of automated deduction [19]. One important reason for that has to be attributed to the fact that Skolem terms can be easily manipulated in deductions. In fact they can be treated (both at the syntactical and at the semantical levels) analogously to the terms already present in the initial language. However, if a tableau system adopts a $\boldsymbol{\delta}$-rule variant based on intricate constructions of Skolem terms, proving its soundness may become problematic [1, 3].

In consideration of that, in [4] we have presented a generic $\boldsymbol{\delta}$-rule whose soundness is characterized by some conditions which can be easily instantiated to prove the correctness of known $\boldsymbol{\delta}$-rule variants based on Skolemization.

Additionally, such a general framework allows one to compare structurally the various $\boldsymbol{\delta}$-rule variants in such a way as to appraise in a natural way their efficiency. It also gives information on the choice of the Skolem symbols and on the construction of Skolem terms, augmented signatures, and canonical models.

An approach alternative to the one based on Skolemization has been used in [11], where the $\boldsymbol{\delta}^{\epsilon}$-rule, adopting Hilbert's ϵ-terms in place of Skolem terms, has been defined.

In order to use ϵ-terms as syntactical entities in the context of tableau proofs, a suitable semantics, called *substitutive*, has been introduced in [11], which makes ϵ-terms sensitive to the syntactical manipulations usually applied in automated deduction, such as the substitution of terms in a formula.

The relationship between ϵ-terms and Skolem terms, when used for the purpose of eliminating existential quantifiers, has been discussed in [11]. In this paper we show how the $\boldsymbol{\delta}^{\epsilon}$-rule can be mapped, and therefore can be studied, in the context of $\boldsymbol{\delta}$-rules based on the Skolemization technique. Our generic $\boldsymbol{\delta}$-rule serves to such purpose.

We start by defining a $\boldsymbol{\delta}$-rule variant, called $\boldsymbol{\delta}^{\mathbf{sk}}$-rule, whose soundness follows immediately from the fact that it is an instance of our generic $\boldsymbol{\delta}$-rule. Then the $\boldsymbol{\delta}^{\mathbf{sk}}$-rule is identified with the $\boldsymbol{\delta}^{\epsilon}$-rule in the context of a same tableau system. This suggests a mechanism for the $\boldsymbol{\delta}^{\epsilon}$-rule to construct substitutive pre-structures from classical first-order structures, similar to the one used by the generic $\boldsymbol{\delta}$-rule to construct canonical structures. As a by-product, we end up with a new soundness proof of the $\boldsymbol{\delta}^{\epsilon}$-rule, alternative to the one presented in [11].

2 Preliminaries

Before going into details, we review some notation and terminology which will be used throughout the paper.

2.1 Signatures and languages

Let $\Sigma = (\mathcal{P}, \mathcal{F})$ be a *signature*, where $\mathcal{P}$ and $\mathcal{F}$ are countable collections of predicate and function symbols, respectively, and let Var be a fixed countable collection of individual variables. Then the *language* $\mathcal{L}_{\Sigma}$ is the collection of all first-order terms and formulae involving besides the standard logical symbols, also individual variables in Var, and predicate and function symbols of the signature Σ. For the sake of simplicity, we will assume that the primitive propositional connectives of $\mathcal{L}_{\Sigma}$ are $\wedge$, $\vee$, and $\neg$. Thus, other connectives will be used just as shorthands: for instance, the formula $\varphi \supset \psi$ is to be considered as a shorthand for $\neg\varphi \vee \psi$.

Terms and formulae construction rules, notions of free and bound variables, of closed formulae (sentences), of (immediate) subformulae, of occurrences of terms and formulae in a given formula are the standard ones. Precise definitions can be found in [10] and in [15].

For any formula φ in the language $\mathcal{L}_\Sigma$, the collection of free variables occurring in φ is denoted by $Free(\varphi)$, whereas the collection of bound variables in φ is denoted by $Bound(\varphi)$.

It is convenient to assume that the individual variables Var are arranged in a sequence $\langle \ldots, \mathsf{x}_{-2}, \mathsf{x}_{-1}, \mathsf{x}_0, \mathsf{x}_1, \mathsf{x}_2, \ldots \rangle$, and that the two subsequences

- $Var^- = \langle \mathsf{x}_{-1}, \mathsf{x}_{-2}, \ldots \rangle$, and
- $Var^+ = \langle \mathsf{x}_0, \mathsf{x}_1, \mathsf{x}_2, \ldots \rangle$

are singled out, to denote bound and free variables, respectively.

In this paper, we will explicitly refer to the variables in Var only in Section 5, where quasi-key formulae are defined, and in Section 6, where the result of isomorphism between tableau proofs applying the $\boldsymbol{\delta}^{\mathbf{sk}}$-rule and tableau proofs with the $\boldsymbol{\delta}^{\epsilon}$-rule is discussed. But when it is not necessary to insist on such a convention, we will just use the meta-variables x, y, z (possibly subscribed with natural numbers) standing for generic variables in Var.

By $\mathcal{L}_\Sigma^+$ we denote the collection of all formulae in $\mathcal{L}_\Sigma$ whose free variables are contained in Var^+ and whose bound variables in Var^-. In addition, for any given formula φ in $\mathcal{L}_\Sigma^+$, by $\mathcal{L}_\varphi^+$ we denote the collection of all formulae in $\mathcal{L}_\Sigma^+$ which mention only predicate and function symbols occurring in φ.

Without loss of generality, throughout the paper we will consider only formulae in $\mathcal{L}_\Sigma^+$.

For every $\varphi \in \mathcal{L}_\Sigma^+$, we denote by $[\varphi]$ the equivalence class of all the formulae which are equal to φ up to renaming of variables (even bound ones).

An occurrence of a subformula in a formula φ is *positive* if it is in the scope of an *even* number of negation symbols.

A *(variable-) substitution* is a mapping $\sigma : Var^+ \rightarrow Terms_\Sigma^+$, where $Terms_\Sigma^+$ is the collection of all terms on $\Sigma \cup Var^+$. The action of a substitution is recursively extended to terms and formulae of $\mathcal{L}_\Sigma^+$ as usual. We say that a substitution σ is *free* for a formula φ, if the formula φ and the formula $\varphi\sigma$, resulting from applying σ to φ, have exactly the same occurrences of bound variables. For instance, the substitution $\sigma = \{x \leftarrow f(y)\}$ is free for $(\forall z)P(z, x)$ whereas $\sigma' = \{x \leftarrow h(z)\}$ is not. Notice that by the distinction we have done between variables in Var^+ and in Var^-, in this paper we deal with free substitutions.

Let Φ be a set of terms or a set of quantifier-free formulae. A substitution σ is called a *unifier* for Φ if $|\Phi\sigma| = 1$. A unifier σ for Φ is a *most general unifier (MGU)* if, for every unifier τ for Φ, there exists a substitution θ such that $\tau = \sigma\theta$. For example, the substitutions $\tau = \{y \leftarrow x, x \leftarrow f(x)\}$ and $\sigma = \{x \leftarrow f(y)\}$ are both unifiers for $\Phi = \{R(x), R(f(y))\}$, but only σ is an MGU.

2.2 Structures and assignments

A *structure* $\boldsymbol{\mathcal{M}} = \langle \boldsymbol{\mathcal{D}}, \boldsymbol{\mathcal{I}} \rangle$ for a signature $\Sigma = (\mathcal{P}, \mathcal{F})$ consists of a nonempty domain $\boldsymbol{\mathcal{D}}$ and an interpretation $\boldsymbol{\mathcal{I}}$ for the function and predicate symbols in

Σ such that $P^{\mathcal{I}} : \mathcal{D}^{\text{arity}(P)} \to \{\mathbf{true}, \mathbf{false}\}$, for every predicate symbol $P \in \mathcal{P}$, and $f^{\mathcal{I}} : \mathcal{D}^{\text{arity}(f)} \to \mathcal{D}$, for every function symbol $f \in \mathcal{F}$.

An *assignment* $\mathcal{A}$ relative to a structure $\mathcal{M} = \langle \mathcal{D}, \mathcal{I} \rangle$ and to a language $\mathcal{L}_\Sigma$ is a mapping $\mathcal{A} : \mathit{Var} \to \mathcal{D}$. An *x-variant* of an assignment $\mathcal{A}$ is an assignment $\mathcal{A}'$ such that $y^{\mathcal{A}'} = y^{\mathcal{A}}$, for every variable y different from x. We use the notation $\mathcal{A}[x \leftarrow \boldsymbol{d}]$ to denote the x-variant of $\mathcal{A}$ such that $x^{\mathcal{A}} = \boldsymbol{d}$, for any $\boldsymbol{d} \in \mathcal{D}$.

The notions of satisfiability and validity of a (set of) formula(e) are the standard ones. So, for instance, given a structure $\mathcal{M} = \langle \mathcal{D}, \mathcal{I} \rangle$ and an assignment $\mathcal{A}$ relative to $\mathcal{M}$, we write $(\mathcal{M}, \mathcal{A}) \models \varphi$ or $\varphi^{\mathcal{I},\mathcal{A}} = \mathbf{true}$ to express the fact that the formula φ is true when its predicate and function symbols are interpreted by $\mathcal{I}$, its free variables are interpreted according to $\mathcal{A}$, and its logical symbols have their standard meaning. The notation $\mathcal{M} \models \varphi$ is used to indicate that $(\mathcal{M}, \mathcal{A}) \models \varphi$, for every assignment $\mathcal{A}$, whereas $\models \varphi$ denotes that $\mathcal{M} \models \varphi$, for every structure $\mathcal{M}$ over the signature of the language (in this case we say that φ is valid). Again, further details are available in [10].

2.3 Unifying notation

For the sake of simplicity we use Smullyan's unifying notation, which has the advantage of being compact, thus cutting down on the number of cases that must be considered in proofs. Smullyan divides the formulae of the language into four categories: conjunctive, disjunctive, universal, and existential formulae (called $\boldsymbol{\alpha}$-, $\boldsymbol{\beta}$-, $\boldsymbol{\gamma}$-, and $\boldsymbol{\delta}$-formulae, respectively). In particular, $\boldsymbol{\delta}$-formulae are those of the form $(\exists x)\varphi$ and $\neg(\forall x)\varphi$, whereas $\boldsymbol{\gamma}$-formulae are those of the form $(\forall x)\varphi$ and $\neg(\exists x)\varphi$.

Given a $\boldsymbol{\delta}$-formula δ, the notation $\delta_0(x)$ will be used to denote the formula φ, if δ is of the form $(\exists x)\varphi$, or to denote the formula $\neg\varphi$, if δ is of the form $\neg(\forall x)\varphi$. In any case, we will refer to $\delta_0(x)$ as *the instance of* δ and to x as *the quantified variable of* δ.

Likewise, for any $\boldsymbol{\gamma}$-formula γ, $\gamma_0(x)$ denotes the formula φ or $\neg\varphi$, according to whether γ has the form $(\forall x)\varphi$ or $\neg(\exists x)\varphi$, respectively.[1]

Let us define the complement operator by putting:

$$\mathsf{C}(X) = \begin{cases} Z & \text{if } X = \neg Z \\ \neg X & \text{otherwise.} \end{cases}$$

Then to each $\boldsymbol{\alpha}$- and $\boldsymbol{\beta}$-formula, one can associate its components as shown in Table 1.

Plainly, the following equivalences hold:

$$\models \alpha \equiv \alpha_1 \wedge \alpha_2\,, \quad \models \beta \equiv \beta_1 \vee \beta_2\,, \quad \models \gamma \equiv (\forall x)\gamma_0(x)\,, \quad \models \delta \equiv (\exists x)\delta_0(x)\,.$$

[1] To simplify the exposition, throughout the paper we will feel free to use the same meta-variable x to represent $(Qx)\varphi$ (where Q is a quantifier) and its instance $\varphi(x)$, despite the fact that they stand for different individual variables of Var. In fact, the occurrences of x in $(Qx)\varphi$ stand for a variable in Var^-, whereas the ones in $\varphi(x)$ stand for a variable in Var^+.

α	α_1	α_2
$X \wedge Y$	X	Y
$\neg(X \vee Y)$	$\mathtt{C}(X)$	$\mathtt{C}(Y)$
$\neg(X \supset Y)$	X	$\mathtt{C}(Y)$
$\neg\neg X$	X	–

β	β_1	β_2
$X \vee Y$	X	Y
$\neg(X \wedge Y)$	$\mathtt{C}(X)$	$\mathtt{C}(Y)$
$(X \supset Y)$	$\mathtt{C}(X)$	Y

Table 1. $\boldsymbol{\alpha}$- and $\boldsymbol{\beta}$-components.

$$\frac{\alpha}{\begin{array}{c}\alpha_1\\ \alpha_2\end{array}} \qquad \frac{\beta}{\beta_1 \mid \beta_2} \qquad \frac{\gamma}{\gamma_0(x)} \qquad \frac{\delta}{\delta_0(f(\overrightarrow{S}))}$$

Table 2. Tableau rules for a generic calculus.

2.4 Free variable semantic tableaux

Tableaux are proof systems based on a systematic decomposition of the (set of) formula(e) to be proved till a manifest contradiction is found. Decomposition is performed by suitable expansion rules, whereas contradictions are detected by a closure rule. Table 2 presents the rules of a generic free variable tableau calculus (we refer the reader to [7] and [12] for a deeper treatment of the tableau method). The $\boldsymbol{\alpha}$-rule is used to expand conjunctive formulae ($\boldsymbol{\alpha}$-formulae). It decomposes a formula α into its components α_1 and α_2 as described in Table 1, giving rise to a single expansion. If α is of type $\neg\neg X$, then α_2 is not present in the $\boldsymbol{\alpha}$-rule.

Analogously, the $\boldsymbol{\beta}$-rule is employed to decompose disjunctive formulae ($\boldsymbol{\beta}$-formulae). The components β_1 and β_2 of a $\boldsymbol{\beta}$-formula β are determined as illustrated in Table 1. Further, the presence of a vertical bar in the schema of the $\boldsymbol{\beta}$-rule indicates that its application originates a branch splitting.

The $\boldsymbol{\gamma}$-rule decomposes universal formulae, instantiating them with new free variables. To speed-up tableau closure, it is convenient to require that the variable x in the $\boldsymbol{\gamma}$-rule be a free variable in Var^+, new to the current tableau.

Expansion of $\boldsymbol{\delta}$-formulae is performed by the $\boldsymbol{\delta}$-rule which consists in instantiating the existentially quantified formula with a Skolem term $f(\overrightarrow{S})$, where f is a function symbol and $\overrightarrow{S}$ an ordered tuple of terms which have to satisfy suitable requirements.

We postpone the precise definition of $\boldsymbol{\delta}$-rule and Skolem terms to Section 3. There we will characterize the proviso of the $\boldsymbol{\delta}$-rule in such a way as to enforce soundness and encompass the $\boldsymbol{\delta}$-rule variants present in the literature which are based on Skolemization. Here we just recall that Skolem terms are constructed using function symbols from a countably infinite set **sko**, disjoint from $\mathcal{F}$ and such that it contains countably many distinct function symbols of any arity. We indicate with $\Sigma_{\mathbf{sko}} = (\mathcal{P}, \mathcal{F} \cup \mathbf{sko})$ the augmented signature.

2.5 Tableau proofs

We represent a tableau $\mathcal{T}$ for a set of closed formulae Φ of $\mathcal{L}^+_\Sigma$ as a dyadic ordered tree whose nodes are labelled with formulae of $\mathcal{L}^+_{\Sigma_{\mathbf{sko}}}$. Any maximal path on $\mathcal{T}$ is a *branch* of $\mathcal{T}$.

Given a set Φ of closed formulae of $\mathcal{L}^+_\Sigma$, a *free variable tableau* for Φ is recursively defined as follows:

- The tree with only one node labelled with **true** is a tableau for Φ (*initial tableau*);
- Let $\mathcal{T}$ be a tableau for Φ, ϑ a branch of $\mathcal{T}$, and φ a formula occurring in $\vartheta \cup \Phi$. Then the tree $\mathcal{T}'$, obtained from $\mathcal{T}$ by extending ϑ through the application of an expansion rule from Table 2 as described by points 1-4 below, is a tableau for Φ.
  1. If φ is an $\boldsymbol{\alpha}$-formula α, the $\boldsymbol{\alpha}$-rule is applied, and the components α_1 and α_2 are appended to ϑ, thus yielding the extended branch $\vartheta; \alpha_1; \alpha_2$.
  2. If φ is a $\boldsymbol{\beta}$-formula β, the $\boldsymbol{\beta}$-rule is applied, and the two extended branches $\vartheta; \beta_1$ and $\vartheta; \beta_2$ are constructed out of ϑ.
  3. If φ is a $\boldsymbol{\gamma}$-formula γ, by the application of the $\boldsymbol{\gamma}$-rule, ϑ is extended to $\vartheta; \gamma_0(x)$, where x is a free variable in Var^+, new to the current tableau.
  4. If φ is a $\boldsymbol{\delta}$-formula δ, the $\boldsymbol{\delta}$-rule is applied, and $\delta_0(f(\overrightarrow{S}))$ is appended to ϑ yielding the extended branch $\vartheta; \delta_0(f(\overrightarrow{S}))$, for a suitable term $f(\overrightarrow{S})$.
- Let $\mathcal{T}$ be a tableau for Φ, ϑ a branch of $\mathcal{T}$, and ψ, ψ' literals on ϑ. If ψ and $\mathsf{C}(\psi')$ are unifiable with *MGU* σ, and $\mathcal{T}' = \mathcal{T}\sigma$ is obtained by applying the substitution σ to all the formulae on $\mathcal{T}$, then $\mathcal{T}'$ is a tableau for Φ.

Given a structure $\boldsymbol{\mathcal{M}} = \langle \boldsymbol{\mathcal{D}}, \boldsymbol{\mathcal{I}} \rangle$ for $\Sigma_{\mathbf{sko}}$ and an assignment $\mathcal{A}$ relative to $\boldsymbol{\mathcal{M}}$, a branch ϑ of a tableau $\mathcal{T}$ is *satisfiable* by $\boldsymbol{\mathcal{M}}$ and $\mathcal{A}$, in which case we write $(\boldsymbol{\mathcal{M}}, \mathcal{A}) \models \vartheta$, if $(\boldsymbol{\mathcal{M}}, \mathcal{A}) \models \varphi$ holds, for each φ in the branch ϑ.

A tableau $\mathcal{T}$ for a set of closed formulae Φ of $\mathcal{L}^+_\Sigma$ is *satisfiable* if there exists a structure $\boldsymbol{\mathcal{M}} = \langle \boldsymbol{\mathcal{D}}, \boldsymbol{\mathcal{I}} \rangle$ for $\Sigma_{\mathbf{sko}}$ such that, for every variable assignment $\mathcal{A}$, $(\boldsymbol{\mathcal{M}}, \mathcal{A}) \models \vartheta$ holds for some branch ϑ of $\mathcal{T}$. In such a case, we say that $\boldsymbol{\mathcal{M}}$ is a *model* of $\mathcal{T}$, and also write $\boldsymbol{\mathcal{M}} \models \mathcal{T}$.

A branch ϑ of a tableau $\mathcal{T}$ is *closed* if it contains two literals ψ and ψ' such that $\psi = \mathsf{C}(\psi')$.

A tableau $\mathcal{T}$ is *closed* if all of its branches are closed.

A *tableau proof* of the unsatisfiability of a set Φ of formulae of $\mathcal{L}^+_\Sigma$ is a closed tableau for Φ. A formula φ is a *theorem* of a tableau calculus if the latter produces a closed tableau for $\{\neg\varphi\}$. A tableau calculus is *sound* if every theorem that it proves is a valid formula, whereas it is *complete* if every valid formula is a theorem of it.

It is useful to index the branches of a tableau as well as the formula occurrences over each of them with incremental natural numbers. In this way the position of the occurrence of a formula φ over a tableau $\mathcal{T}$ is uniquely determined by a pair (n, m) where n is the index of ϑ, the branch of $\mathcal{T}$ where the occurrence of φ lies, and m is the index of the occurrence of φ over ϑ.

Let $\mathcal{C}_1$ and $\mathcal{C}_2$ be variants of the generic tableau calculus introduced above, $\mathcal{T}_1$ a tableau relative to $\mathcal{C}_1$ over the language $\mathcal{L}^+_{\Sigma_1}$, and $\mathcal{T}_2$ a tableau relative to $\mathcal{C}_2$ over the language $\mathcal{L}^+_{\Sigma_2}$.

$\mathcal{T}_1$ is *isomorphic* to $\mathcal{T}_2$ if there exists a *bijection* $\mathsf{f} : \mathcal{L}^+_{\Sigma_1} \to \mathcal{L}^+_{\Sigma_2}$ such that

- $\mathcal{T}_1$ is an initial tableau for a set Φ of closed formulae of $\mathcal{L}^+_{\Sigma_1}$, *if and only* if $\mathcal{T}_2$ is an initial tableau for the set of closed formulae $\Phi' = \{\mathsf{f}(\varphi) : \varphi \in \Phi\}$;
- If $\bar{\mathcal{T}}_1$ is a tableau for Φ relative to $\mathcal{C}_1$ isomorphic to the tableau $\bar{\mathcal{T}}_2$ for Φ' relative to $\mathcal{C}_2$, if ϑ_1 is the branch of index n on $\bar{\mathcal{T}}_1$, and φ the formula occurrence of index m on $\vartheta_1 \cup \Phi$,

  then, $\mathcal{T}_1$ is obtained from $\bar{\mathcal{T}}_1$ by extending ϑ_1 through the application of an expansion rule from the calculus $\mathcal{C}_1$, *if and only if* $\mathcal{T}_2$ is obtained from $\bar{\mathcal{T}}_2$ by extending its branch of index n, ϑ_2, through the application of an expansion rule from $\mathcal{C}_2$, as described in the following.

  1. if φ is an $\boldsymbol{\alpha}$-formula α, the rule of $\mathcal{C}_1$ yielding the extended branch $\vartheta_1; \alpha_1; \alpha_2$ is applied to $\bar{\mathcal{T}}_1$ *if and only if* the corresponding rule of $\mathcal{C}_2$ yielding the extended branch $\vartheta_2; \mathsf{f}(\alpha_1); \mathsf{f}(\alpha_2)$ is applied to $\bar{\mathcal{T}}_2$.
  2. if φ is a $\boldsymbol{\beta}$-formula β, the rule of $\mathcal{C}_1$ constructing the extended branches $\vartheta_1; \beta_1$ and $\vartheta_1; \beta_2$ out of ϑ_1, is applied to $\bar{\mathcal{T}}_1$ *if and only if* the corresponding rule of $\mathcal{C}_2$, yielding the branches $\vartheta_2; \mathsf{f}(\beta_1)$ and $\vartheta_2; f(\beta_2)$ out of ϑ_2, is applied to $\bar{\mathcal{T}}_2$.
  3. the cases where φ is either a $\boldsymbol{\gamma}$- or a $\boldsymbol{\delta}$-formula are treated in a similar way.

- If $\bar{\mathcal{T}}_1$ is a tableau for Φ relative to $\mathcal{C}_1$ isomorphic to the tableau $\bar{\mathcal{T}}_2$ for Φ' relative to $\mathcal{C}_2$, if ϑ_1 is the branch of index n on $\bar{\mathcal{T}}_1$ and ψ, ψ' literals on ϑ_1 of indexes m and m', respectively, such that ψ and $\mathsf{C}(\psi')$ are unifiable with MGU σ,

  then, $\mathcal{T}_1 = \bar{\mathcal{T}}_1\sigma$, *if and only if* $\mathcal{T}_2 = \bar{\mathcal{T}}_2\mathsf{f}(\sigma)$ where $\mathsf{f}(\sigma)$ is an MGU of $\mathsf{f}(\psi)$, $\mathsf{C}(\mathsf{f}(\psi'))$, and $\mathsf{f}(\psi)$, $\mathsf{f}(\psi')$ are the literals of indexes m and m' on the branch ϑ_2 of index n on $\bar{\mathcal{T}}_2$. Notice that if $\sigma = \{x_1 \leftarrow t_1, \ldots, x_n \leftarrow t_n\}$, with $\mathsf{f}(\sigma)$ we indicate the substitution $\{\mathsf{f}(x_1) \leftarrow \mathsf{f}(t_1), \ldots, \mathsf{f}(x_n) \leftarrow \mathsf{f}(t_n)\}$.

3 A generic sound δ-rule for Skolemization based δ-rules

In order to show that a **δ**-rule variant is sound, one has to prove that it preserves the semantics of existential quantification in tableau proofs.

Thus, soundness of a **δ**-rule variant would be trivially verified if the effect of its application over the **δ**-formulae occurring on a tableau $\mathcal{T}$ could be identified with their Skolemization, since, as is well-known, Skolemization preserves the semantics of existential quantification [15].

Definition 1 (Skolemization). *Let Φ be a set of formulae containing a formula $\varphi = (\exists x)\delta_0(x)$, whose free variables are among $x_1, \ldots, x_n$. If f is an n-ary function symbol not occurring in Φ, then the formula $\delta_0(x \leftarrow f(x_1, \ldots, x_n))$ is called a* Skolemization *of φ with respect to Φ.* □

However, several **δ**-rule variants in the literature do not lend themselves to such a direct identification. For instance, some of them allow to reuse the same Skolem symbol in a proof [2]. Others, in addition to the previous feature, present the characteristic of producing a Skolem term containing only a proper subset of the free variables in the formula to be Skolemized [1, 3].

Example 1. Let $\delta_1 = (\exists x)P(x, x_1)$ and $\delta_2 = (\exists x)P(x, x_2)$ be two **δ**-formulae occurring on a same branch of a tableau based on the $\boldsymbol{\delta}^{++}$-rule [2]. Then δ_1 and δ_2 are expanded with the Skolem terms $f_{[\delta_1]}(x_1)$ and $f_{[\delta_2]}(x_2)$, respectively. But since the $\boldsymbol{\delta}^{++}$-rule proviso assigns the same Skolem function symbol to formulae identical up to variable renaming, the symbols $f_{[\delta_1]}$ and $f_{[\delta_2]}$ coincide.

Let us suppose that δ_1 is processed before δ_2. Then the expansion of δ_2 is not a valid Skolemization of δ_2 with respect to the formulae occurring on the tableau, since it uses as Skolem symbol a symbol already present in the proof.

□

Example 2. Let $\delta_1 = (\exists x)(P(x, x_1) \wedge Q(x_2, g(x_3)))$ be a **δ**-formula occurring on a tableau constructed according to the $\boldsymbol{\delta}^{*}$-rule [1]. The Skolem term provided by the $\boldsymbol{\delta}^{*}$-proviso to instantiate δ is $f_{[\delta]}(x_1)$ (the interested reader is referred to [1] or to [4] for the details of the construction). Since the variables x_2 and x_3, which occur both free in δ, are not present among the arguments of the term $f_{[\delta]}(x_1)$, such expansion is not a valid Skolemization of δ with respect to any set of formulae Φ containing δ. □

Cases like the one described in Example 1 can be treated by constructing the Skolemization of a **δ**-formula δ with respect to the set $\Phi = \{\delta\}$ rather than with respect to the set of all the formulae in the current tableau. This choice does not penalize more traditional variants of the **δ**-rule (such as Fitting's **δ**-rule [10]

$$\delta \xrightarrow{\;\Theta_1\;} \xi(\delta^s)\sigma \xrightarrow{\;\Theta_2\;} \xi(\delta^s_0(f(\overrightarrow{H})))\sigma \xrightarrow{\;\Theta_3\;} \delta_0(f(\overrightarrow{S}))$$

Fig. 1. Characterization of a generic sound **δ**-rule as a series of satisfiability preserving transformations.

and the $\boldsymbol{\delta}^{+}$-rule [13]) that require the introduction of a Skolem symbol new to the current tableau at each rule application. In fact it is possible to equip every **δ**-formula of the language with a countably infinite number of Skolem functions to be "consumed" in the course of the proof.

As witnessed by Example 2, the expedient of constructing the Skolemization of a **δ**-formula δ with respect to $\Phi = \{\delta\}$ is not enough to guarantee the applicability of the Skolemization to prove the soundness of the generic **δ**-rule. In fact the employment of **δ**-rules like the ones presented in [1] and in [4] not always returns the Skolemization of the considered formula. It turns out that Skolemization must be restricted only to a subset of the **δ**-formulae (called *Skolemizable* formulae) whereas the **δ**-formulae that are not Skolemizable have to be related to their corresponding Skolemizable formula through suitable satisfiability preserving transformations. In the following, Skolemizable formulae are denoted with δ^s.

According to these considerations, in [4] we have defined a generic **δ**-rule of the form shown in Table 2, whose soundness can be shown through a series of *satisfiability preserving* transformations, as depicted in Figure 1. We devote the rest of this section to the presentation of such result.

Assuming that our **δ**-rule instantiates a given **δ**-formula δ to $\delta_0(f(\overrightarrow{S}))$, then in Figure 1 we have pointed out three such transformations, Θ_1, Θ_2, and Θ_3 (notice that it may be convenient to further split some of them into more basic ones, as will be seen in Section 3.1). In particular,

- Θ_1 transforms the initial **δ**-formula δ into an intermediate formula $\xi(\delta^s)\sigma$, where δ^s is Skolemizable and occurs only positively in ξ;
- Θ_2 transforms $\xi(\delta^s)\sigma$ into $\xi(\delta^s_0(f(\overrightarrow{H})))\sigma$, where $\overrightarrow{H}$ is a tuple of variables and $\delta^s_0(f(\overrightarrow{H}))$ is a Skolemization of δ^s with respect to $\Phi = \{\delta^s\}$;
- Θ_3 transforms $\xi(\delta^s_0(f(\overrightarrow{H})))\sigma$ into $\delta_0(f(\overrightarrow{S}))$, where $\overrightarrow{S} = \overrightarrow{H}\sigma$.

Thus, with the above approach in mind, we only need to define precisely such satisfiability preserving transformations, which will be done next.

3.1 Skolem terms construction rule

The Skolem term $f(\overrightarrow{S})$ in the $\boldsymbol{\delta}$-rule in Table 2 consists of a function symbol $f \in \mathbf{sko}$ of arity $n \geq 0$ and an n-tuple $\overrightarrow{S}$ of terms in $\mathcal{L}^+_{\Sigma_{\mathbf{sko}}}$, whose variables belong to Var^+. In general, the constraints that $f(\overrightarrow{S})$ must satisfy may depend on the current tableau $\mathcal{T}$, on the branch ϑ which is about to be expanded, and on the $\boldsymbol{\delta}$-formula δ on ϑ that is about to be instantiated. Notice that the branch ϑ can be encoded by its index in the tableau $\mathcal{T}$ and that the formula δ can be encoded by its position on ϑ. Therefore, if we denote by $Tab^+_{\Sigma_{\mathbf{sko}}}$ the collection of all tableaux on $\mathcal{L}^+_{\Sigma_{\mathbf{sko}}}$, we can assume that the Skolem term $f(\overrightarrow{S})$ is computed by a certain function

$$\mathcal{S}_{\boldsymbol{\delta}} : Tab^+_{\Sigma_{\mathbf{sko}}} \times \mathbb{N} \times \mathbb{N} \to Terms^+_{\Sigma_{\mathbf{sko}}} \, ,$$

called *Skolem terms construction rule.*

Next we state how to characterize the Skolem terms construction rule $S_{\boldsymbol{\delta}}$ in order to obtain a sound calculus. We introduce some conditions that guarantee the construction of Skolem terms so as to preserve not only the satisfiability of the $\boldsymbol{\delta}$-formula occurrence which is to be expanded, but also that of the whole tableau. In particular condition **C**0 below describes how the objects needed to construct the Skolem terms are generated, whereas conditions **C**1-**C**7 state the relations that must hold between such objects.

C0. We will assume that the $\boldsymbol{\delta}$-rule proviso provides a fixed collection Δ^s of *Skolemizable* $\boldsymbol{\delta}$-formulae of $\mathcal{L}^+_{\Sigma_{\mathbf{sko}}}$, closed under renaming of variables (even bound ones). In other words, we require that if $\delta^s \in \Delta^s$, then $[\delta^s] \subseteq \Delta^s$.
We will also assume that for each $\delta^s \in \Delta^s$, the $\boldsymbol{\delta}$-rule proviso provides a nonempty set of function symbols $\mathbf{sko}_{[\delta^s]} \subseteq \mathbf{sko}$ such that $\mathbf{sko}_{[\delta^s]}$ contains no function symbol occurring in δ^s and $\text{arity}(f) \geq |Free(\delta^s)|$, for each $f \in \mathbf{sko}_{[\delta^s]}$. Additionally, we will require that the sets $\mathbf{sko}_{[\delta^s]}$ are pairwise disjoint and form a partition of $\mathbf{sko}$. In this way, only formulae that are syntactically identical up to variable renaming may share the same function symbol.
Let δ be a $\boldsymbol{\delta}$-formula at position n in the m-th branch ϑ of a tableau $\mathcal{T}$, which we intend to expand. In order to define $\mathcal{S}_{\boldsymbol{\delta}}(\mathcal{T}, m, n)$, we require that the $\boldsymbol{\delta}$-rule proviso associates to such occurrence of δ the following objects:
- a $\boldsymbol{\delta}$-formula δ^a in the language $\mathcal{L}^+_{\delta}$, called *abstraction formula of* δ, and a substitution σ;
- a $\boldsymbol{\delta}$-formula δ^s and a formula $\xi = \xi(\delta^s)$ in the language $\mathcal{L}^+_{\delta}$, respectively called *Skolemization formula* and *transformation formula of* δ;[2]

[2] Given two first-order formulae φ and ψ, we write $\varphi = \varphi(\psi)$ to indicate that the occurrences of ψ in φ play a significant role. Thus, for instance, if ψ' is another formula, by $\varphi(\psi')$ we denote the formula resulting from φ when all occurrences of ψ are replaced by ψ'.

– a function symbol $f \in \mathbf{sko}_{[\delta^s]}$ and an ordered tuple $\overrightarrow{H}$ of variables in *Var* containing all the free variables occurring in δ^s and such that $\text{arity}(f) = |\overrightarrow{H}|$.

The formula δ^a and the substitution σ allow to take care of those $\boldsymbol{\delta}$-rule variants which assign the same function symbol to $\boldsymbol{\delta}$-formulae which are identical up to substitutions [2, 1, 3]. δ^a represents the most general formula with respect to the substitution σ, as specified by the proviso of our generic $\boldsymbol{\delta}$-rule.
Formulae δ^s and ξ allow to encompass those $\boldsymbol{\delta}$-rule variants that construct Skolem terms using only parts of δ^a [1, 3]: the formula ξ has the function of preserving satisfiability in the transformation, whereas δ^s is the formula which is actually Skolemized. $f(\overrightarrow{H})$ is the term used to skolemize δ^s.

We will further assume that such objects satisfy the following conditions:

C1. the substitution σ must be free for the formulae δ^a, $\delta^a_0(f(\overrightarrow{H}))$, $\xi(\delta^s)$, and $\xi(\delta^s_0(f(\overrightarrow{H})))$;

C2. δ^s is a subformula of ξ, occurring *only* positively in it;

C3. the quantified variable of δ^s is the same as the one of δ^a and occurs in ξ only within occurrences of δ^s;

C4. $\models \delta \supset \delta^a\sigma\,;$

C5. $\models \delta^a_0(x_0) \supset \xi(\delta^s_0(x_0))\,;$

C6. $\models \delta^a \supset (\forall x_0)(\xi(\delta^s_0(x_0)) \supset \delta^a_0(x_0))\,;$

C7. $\models (\delta^a_0(f(\overrightarrow{H})))\sigma \supset \delta_0(f(\overrightarrow{H})\sigma)\,.$

Then we put:

$$\mathcal{S}_{\boldsymbol{\delta}}(\mathcal{T}, m, n) =_{Def} f(\overrightarrow{H})\sigma\,. \tag{1}$$

Soundness of the tableau calculus described in Section 2.4, whose associated Skolem terms construction rule satisfies the above conditions **C**0-**C**7, is proved by showing that tableau satisfiability is preserved by the expansion rules in Table 2 and substitution applications. To this purpose, it is convenient to stratify the language $\mathcal{L}^+_{\Sigma_{\mathbf{sko}}}$, and then show how we can expand a given structure for $\mathcal{L}^+_{\Sigma}$ to a canonical structure for $\mathcal{L}^+_{\Sigma_{\mathbf{sko}}}$.

3.2 Stratification of the language $\mathcal{L}^+_{\Sigma_{\mathbf{sko}}}$

Let $\Sigma = (\mathcal{P}, \mathcal{F})$ be the initial signature of our language. Then we put: $\bar{\mathcal{F}}_0 =_{Def} \mathcal{F}$ and $\Sigma_0 =_{Def} \Sigma$. Moreover, following [2], for $i \geq 1$ we put recursively

$$\begin{aligned} \bar{\mathcal{F}}_i &=_{Def} \textstyle\bigcup_{\delta^s \in \Delta^s \cap \mathcal{L}^+_{\Sigma_{i-1}}} \mathbf{sko}_{[\delta^s]} \\ \Sigma_i &=_{Def} (\mathcal{P}, \bar{\mathcal{F}}_i)\,. \end{aligned}$$

Without loss of generality, we may assume that $\bigcup_{i=1}^{\infty} \bar{\mathcal{F}}_i = \mathbf{sko}$.[3] Thus we have: $\Sigma_{\mathbf{sko}} = (\mathcal{P}, \bigcup_{i=0}^{\infty} \bar{\mathcal{F}}_i)$.

Relatively to the above stratification of $\Sigma_{\mathbf{sko}}$, it is useful to introduce the following notion of *rank*, for each formula ψ in the language $\mathcal{L}^{+}_{\Sigma_{\mathbf{sko}}}$:

$$rank(\psi) =_{Def} \min\{k \in \mathbb{N} : \psi \text{ is in the language } \mathcal{L}^{+}_{\Sigma_k}\}.$$

3.3 Construction of canonical structures for $\mathcal{L}^{+}_{\Sigma_{\mathbf{sko}}}$

Let $\boldsymbol{\mathcal{M}} = \langle \boldsymbol{\mathcal{D}}, \boldsymbol{\mathcal{I}} \rangle$ be a structure for our initial language $\mathcal{L}^{+}_{\Sigma}$. Let us put $\boldsymbol{\mathcal{I}}_0 =_{Def} \boldsymbol{\mathcal{I}}$ and $\boldsymbol{\mathcal{M}}_0 =_{Def} \boldsymbol{\mathcal{M}}$. Following [2], for $i \geq 1$ we recursively define the expanded structure $\boldsymbol{\mathcal{M}}_i = \langle \boldsymbol{\mathcal{D}}, \boldsymbol{\mathcal{I}}_i \rangle$ for $\mathcal{L}^{+}_{\Sigma_i}$ as follows.

For each predicate symbol $P \in \mathcal{P}$ we put $P^{\boldsymbol{\mathcal{I}}_i} =_{Def} P^{\boldsymbol{\mathcal{I}}_{i-1}}$. Likewise, for each function symbol $f \in \bigcup_{j=0}^{i-1} \bar{\mathcal{F}}_j$ we put $f^{\boldsymbol{\mathcal{I}}_i} =_{Def} f^{\boldsymbol{\mathcal{I}}_{i-1}}$.

Finally, for each function symbol $f \in \bar{\mathcal{F}}_i \setminus \bigcup_{j=0}^{i-1} \bar{\mathcal{F}}_j$, we define $f^{\boldsymbol{\mathcal{I}}_i}$ as follows. Let us assume that $f \in \mathbf{sko}_{[\delta^s]}$, for a certain $\boldsymbol{\delta}$-formula $\delta^s \in \Delta^s \cap \mathcal{L}^{+}_{\Sigma_{i-1}}$, let $k = \text{arity}(f)$, let $\overrightarrow{H}$ be a fixed ordered k-tuple of distinct variables containing all the free variables in δ^s, and let $\overrightarrow{\boldsymbol{b}} \in \boldsymbol{\mathcal{D}}^k$.

We distinguish the following two cases:

(a) if $(\boldsymbol{\mathcal{M}}_{i-1}, \mathcal{A}) \models \delta^s$, for some assignment $\mathcal{A}$ such that $\overrightarrow{H}^{\mathcal{A}} = \overrightarrow{\boldsymbol{b}}$, we put

$$f^{\boldsymbol{\mathcal{I}}_i}(\overrightarrow{\boldsymbol{b}}) =_{Def} \boldsymbol{c} \ , \tag{2}$$

for some $\boldsymbol{c} \in \boldsymbol{\mathcal{D}}$ such that $(\boldsymbol{\mathcal{M}}_{i-1}, \mathcal{A}[x_0 \leftarrow \boldsymbol{c}]) \models \delta^s_0(x_0)$;

(b) otherwise, we put

$$f^{\boldsymbol{\mathcal{I}}_i}(\overrightarrow{\boldsymbol{b}}) =_{Def} \boldsymbol{d} \ , \tag{3}$$

for an arbitrary $\boldsymbol{d} \in \boldsymbol{\mathcal{D}}$.

Finally, we define $\boldsymbol{\mathcal{M}}_{\mathbf{sko}} = \langle \boldsymbol{\mathcal{D}}, \boldsymbol{\mathcal{I}}_{\mathbf{sko}} \rangle$, where $\boldsymbol{\mathcal{I}}_{\mathbf{sko}}|_{\mathcal{P} \cup \mathcal{F}} = \boldsymbol{\mathcal{I}}_0|_{\mathcal{P} \cup \mathcal{F}}$, and $\boldsymbol{\mathcal{I}}_{\mathbf{sko}}|_{\bar{\mathcal{F}}_i} = \boldsymbol{\mathcal{I}}_i|_{\bar{\mathcal{F}}_i}$, for every $i \geq 1$.

Then the soundness of the tableau calculus introduced in Section 2.4, provided that the Skolem terms construction rule is defined as in (1) and conditions **C**0-**C**7 hold, is plainly entailed by the following theorem, whose proof can be found in [4].

Theorem 1. *Assume that the Skolem terms construction rule is defined by (1) and that conditions **C0-C7** hold. Let Φ be a set of closed formulae of $\mathcal{L}^{+}_{\Sigma}$ and let $\boldsymbol{\mathcal{M}} = \langle \boldsymbol{\mathcal{D}}, \boldsymbol{\mathcal{I}} \rangle$ be a structure for $\mathcal{L}^{+}_{\Sigma}$ such that $\boldsymbol{\mathcal{M}} \models \Phi$. Then $\boldsymbol{\mathcal{M}}_{\mathbf{sko}}$ satisfies any tableau for Φ constructed by the calculus in Section 2.4.*[4]

[3] If $\bigcup_{i=1}^{\infty} \bar{\mathcal{F}}_i \subsetneq \mathbf{sko}$, then we can replace **sko** by $\bigcup_{i=1}^{\infty} \bar{\mathcal{F}}_i$ in our previous discussion, by also shrinking Δ^s accordingly.

[4] We recall that **sko** and $\boldsymbol{\mathcal{M}}_{\mathbf{sko}}$ have been defined in Sections 2.4 and 3.3, respectively.

4 Hilbert's ϵ-symbol and the δ^{ϵ}-rule

The ϵ-symbol is an operator for the formation of terms that replaces quantifiers in standard predicative logic. ϵ-terms have the shape $\epsilon x.\varphi$ where x is a variable and φ a formula. An ϵ-term $\epsilon x.\varphi$ is interpreted as *an element of the domain satisfying φ if such an element does exist, an arbitrary element of the domain otherwise*.

We refer the reader to [14], for a thorough account of Hilbert's work on the ϵ-symbol, and to [16, 17], for further elaborations and interpretations in the light of more recent developments of mathematical logic.

Apart from foundational issues (ϵ-substitution method, Hilbert's ϵ-theorems, etc.), the ϵ-symbol has also been applied in fields like linguistic, philosophy, and non-classical logics (see for instance [9, 23]).

The use of the ϵ-symbol in automated deduction is rather limited. It has been introduced in the theorem prover Isabelle [18] and, more recently, in the proof verifier AetnaNova [20]. But, to our knowledge, the only effort to use ϵ-terms as syntactical structures to be applied in the context of a standard tableau calculus (with the purpose of effectively substitute Skolem terms in the proofs) is the one reported in [11].

In this section we review the δ^{ϵ}-rule, the δ-rule variant defined in [11], that expands existentially quantified formulae with ϵ-terms.

Let us introduce some preliminary notions. Let Σ be a signature defined as in Section 2.1. The introduction of the Hilbert's ϵ-symbol among the standard logical symbols results in an enrichment of the language. The rules of formation of terms that are not ϵ-terms and of formulae are the standard ones. So, for instance if f is a symbol in $\mathcal{F}$ of arity $n \geq 0$ and $t_1, \ldots, t_n$ are terms, $f(t_1, \ldots, t_n)$ is a term, and if φ and ψ are formulae, and $\circ$ a binary propositional connective, $\varphi \circ \psi$ is a formula. A totally different discourse holds for ϵ-terms which, in fact, are constructed out of formulae: if φ is a formula and x a variable, then $\epsilon x.\varphi$ is a term. Since ϵ-terms can be used in turn to form other terms and formulae, the ϵ-symbol triggers a mutual recursion between formulae and terms (see [14] for details). We call the resulting language $\mathcal{L}_{\Sigma_\epsilon}$. In this context, notions introduced in Section 2.1 (as, for instance, bound and free variables, substitutions and MGUs) are defined in a similar way, with the only difference that terms are allowed to contain bound variables. Notice in fact that the variable x in the ϵ-term $\epsilon x.\varphi$ is bound.

Even in the present case, assuming that variables in Var^- and Var^+ are employed to denote bound and free variables respectively, we indicate by $\mathcal{L}^+_{\Sigma_\epsilon}$ the collection of all formulae in $\mathcal{L}_{\Sigma_\epsilon}$ whose free variables are in Var^+ and bound variables are in Var^-.

Semantics of $\mathcal{L}^+_{\Sigma_\epsilon}$ is defined, according to [11], through the concept of *pre-structure*. A pre-structure is a triple $\mathfrak{M} = \langle \boldsymbol{\mathcal{D}}, \boldsymbol{\mathcal{I}}, \boldsymbol{\epsilon\text{-}val} \rangle$, where $\boldsymbol{\mathcal{D}}$ and $\boldsymbol{\mathcal{I}}$ are respectively the domain and the interpretation of a structure $\boldsymbol{\mathcal{M}}$ (see Section 2.2) and $\boldsymbol{\epsilon\text{-}val}$ is a function mapping any ϵ-term $\epsilon x.\varphi$ and variable assignment $\mathcal{A}$ into an element $\boldsymbol{d}$ of the domain.

The evaluation of terms and formulae in a pre-structure $\mathfrak{M}$ is defined as for the case of structures, except for the evaluation of ϵ-terms, where we put $\epsilon x.\varphi^{\boldsymbol{\mathcal{I}},\boldsymbol{\epsilon\text{-}val},\mathcal{A}} \equiv_{Def} \boldsymbol{\epsilon\text{-}val}(\epsilon x.\varphi, \mathcal{A})$. In what follows, we write $\langle \boldsymbol{\mathcal{D}}, \boldsymbol{\mathcal{I}}, \boldsymbol{\epsilon\text{-}val}\rangle, \mathcal{A} \models \varphi$ to indicate that the pre-structure $\mathfrak{M} = \langle \boldsymbol{\mathcal{D}}, \boldsymbol{\mathcal{I}}, \boldsymbol{\epsilon\text{-}val}\rangle$ and the assignment $\mathcal{A}$ satisfy φ.

The $\boldsymbol{\delta^{\epsilon}}$-rule proviso. The $\boldsymbol{\delta}^{\epsilon}$-rule is formulated as follows:

$$\frac{\delta}{\delta_0(\epsilon x.\delta_0(x))},$$

where x is the quantified variable in δ and $\epsilon x.\delta_0(x)$ is the ϵ-term associated to δ.

In [11] some restrictions have been applied to the definition of $\boldsymbol{\epsilon\text{-}val}$ with the purpose of providing the ϵ-symbol with a semantics more suitable for the application in automated deduction. Essentially, two different semantics have been introduced in [11]: the *substitutive* semantics, suitable for automated theorem proving, and the *extensional* one, effectively applicable only in interactive theorem provers and requiring the introduction of an additional rule (the ϵ-rule) in the calculus.

Here, we focus on the substitutive semantics, captured by *substitutive pre-structures*, defined as follows.

Definition 2 (Substitutive structure). *A pre-structure $\mathfrak{M}$ for Σ is called* substitutive *if*

1. *given an ϵ-term $\epsilon x.\varphi$ of $\mathcal{L}^+_{\Sigma_\epsilon}$ and two assignments $\mathcal{A}_1, \mathcal{A}_2$ such that $\mathcal{A}_1 \mid_{Free(\exists x\varphi)} = \mathcal{A}_2 \mid_{Free(\exists x\varphi)}$, then $\boldsymbol{\epsilon\text{-}val}(\epsilon x.\varphi, \mathcal{A}_1) = \boldsymbol{\epsilon\text{-}val}(\epsilon x.\varphi, \mathcal{A}_2)$;*
2. *for $y \in Var^+$, $\varphi \in \mathcal{L}^+_{\Sigma_\epsilon}$, assignment $\mathcal{A}$, and term t such that $Free(t) \cap Bound(\epsilon x.\varphi) = \emptyset$, we have*
$$\boldsymbol{\epsilon\text{-}val}(\epsilon x.\varphi\{y \leftarrow t\}, \mathcal{A}) = \boldsymbol{\epsilon\text{-}val}(\epsilon x.\varphi, \mathcal{A}[y \leftarrow t^{\boldsymbol{\mathcal{I}},\boldsymbol{\epsilon\text{-}val},\mathcal{A}}]);$$
3. *for any assignment $\mathcal{A}$ and $\varphi \in \mathcal{L}^+_{\Sigma_\epsilon}$, if $\mathfrak{M}, \mathcal{A} \models (\exists x)\varphi$, then*
$$\mathfrak{M}, \mathcal{A}[x \leftarrow \boldsymbol{\epsilon\text{-}val}(\epsilon x.\varphi, \mathcal{A})] \models \varphi;$$

□

Intuitively, the first condition states that the evaluation of an ϵ-term should depend only on the evaluation of the variables occurring free in it. The second condition expresses the *substitutivity property*, which is very important for the construction of a calculus, as it reflects at the semantic level the syntactical action of substituting a term in a formula. For instance, it allows to infer $Q(\epsilon y.P(a, y))$ from $(\forall x)Q(\epsilon y.P(x, y))$. Finally, the third condition says that an ϵ-term $\epsilon x.\varphi$ should denote an element of the domain that, assigned to the variable x, satisfies φ.

5 The $\boldsymbol{\delta}^{\mathbf{sk}}$-rule

In this section we introduce the $\boldsymbol{\delta}^{\mathbf{sk}}$-rule, a variant of the $\boldsymbol{\delta}^{*^*}$-rule, which was first presented in [3] and then revised in [4]. As the $\boldsymbol{\delta}^{*^*}$-rule, it is based on the *global Skolemization* technique, described in [8] and [6]. The $\boldsymbol{\delta}^{\mathbf{sk}}$-rule proviso does not make use of the concept of relevance and replaces the notion of key formula with the slightly less general one of *quasi-key formula*.

Analogously to key formulae, quasi-key formulae are the formulae of the language most general with respect to substitutions. In particular, they are the only formulae deserving their own Skolem function symbol. Further, to each formula of the language there corresponds a unique (quasi-)key formula. But while between a formula φ and its key formula φ' a relationship holds such that $\varphi = \varphi'\sigma$ up to bounded variables renaming (where σ is a suitable substitution), the relationship between φ and its quasi-key formula φ_1 is simply $\varphi = \varphi_1\sigma$. Prior to formally define quasi-key formulae, we introduce the notion of *quasi-canonical formula*.

Definition 3. *A formula φ is said to be* quasi-canonical *(with respect to the variable x_0) if there is an $n \geq 0$ such that $Free(\varphi) \setminus \{\mathsf{x}_0\} = \{\mathsf{x}_1, \ldots, \mathsf{x}_n\}$, each of these variables occurs in φ just once, and they occur in φ in the order $\mathsf{x}_1, \ldots, \mathsf{x}_n$, from left to right.* □

Every formula φ can be canonized with respect to a designated variable $x \in Var^+$, in the sense that there exists a unique corresponding canonical formula φ_2, such that φ and $\varphi_2\sigma$ are equal, where σ is a substitution free for φ which maps variables into variables and such that $x = \mathsf{x}_0\sigma$.

Example 3. The quasi-canonical formula with respect to x corresponding to the formula $\varphi = (\exists y)(\exists z)(R(x, f(y), z, h(w, w)) \wedge Q(u, v))$ is

$$\varphi_2 = (\exists y)(\exists z)(R(\mathsf{x}_0, f(y), z, h(\mathsf{x}_1, \mathsf{x}_2)) \wedge Q(\mathsf{x}_3, \mathsf{x}_4)).$$

In this case $\sigma = \{\mathsf{x}_0 \leftarrow x, \mathsf{x}_1 \leftarrow w,\ \mathsf{x}_2 \leftarrow w,\ \mathsf{x}_3 \leftarrow u,\ \mathsf{x}_4 \leftarrow u\}$. □

We define quasi-key formulae to be quasi-canonical formulae that are most general with respect to substitutions.

Definition 4. *A formula φ is said to be a* quasi-key formula *if*

– *it is quasi-canonical with respect to* x_0*, and*
– *for all the formulae* ψ *that are quasi-canonical with respect to* x_0*, if there is a substitution* σ *which is free for* ψ *and such that* $\varphi = \psi\sigma$*, then* $\psi = \varphi$*.* □

To any formula of the language there uniquely corresponds a quasi-key formula, as the following lemma states. The proof is along the lines of the one given in [3] for key formulae.

Lemma 1. *Let* φ *be a formula in the language* $\mathcal{L}^+_\Sigma$ *and let* $x \in Var^+$ *be any variable. Then there exists a* unique *quasi-key formula* φ_1 *with respect to* x*, denoted by* $QKey(\varphi, x)$*, and a nonempty collection of substitutions free for* φ_1*, denoted by* $SubstKey(\varphi, x)$*, such that for each* $\sigma \in SubstKey(\varphi, x)$ *we have*

– φ *and* $\varphi_1\sigma$ *are identical, and*
– x *does not occur in* $x_i\sigma$*, for* $x \neq x_i$*.*

□

Example 4. We continue from Example 3. The quasi-key formula with respect to x corresponding to

$$\varphi = (\exists y)(\exists z)(R(x, f(y), z, h(w, w)) \wedge Q(u, v))$$

is

$$\varphi_1 = (\exists y)(\exists z)(R(\mathsf{x}_0, f(y), z, \mathsf{x}_1) \wedge Q(\mathsf{x}_2, \mathsf{x}_3)).$$

A substitution satisfying the conditions of the above lemma is $\sigma' = \{\mathsf{x}_0 \leftarrow x, \mathsf{x}_1 \leftarrow h(w, w,), \mathsf{x}_2 \leftarrow u, \mathsf{x}_3 \leftarrow v\}$. □

The $\boldsymbol{\delta}^{\mathbf{sk}}$-rule proviso.

Definition 5. *Let* δ *be a* $\boldsymbol{\delta}$*-formula of the language* $\mathcal{L}^+_\Sigma$*. Let* $\varphi_1 = QKey(\delta_0(x), x)$ *and let* $\sigma \in SubstKey(\delta_0(x), x)$*, where* x *is the quantified variable of* δ *and let* $S_{\varphi_1} = Free(\varphi_1)\backslash\{x\}$*. Then the* $\boldsymbol{\delta}^{\mathbf{sk}}$*-rule can be schematically described as follows:*

$$\frac{\delta}{\delta_0(h_{\varphi_1}(\overrightarrow{S}_{\varphi_1})\sigma)} \tag{4}$$

□

The $\boldsymbol{\delta}^{\mathbf{sk}}$-rule is sound. Soundness of the $\boldsymbol{\delta}^{\mathbf{sk}}$-rule is proved by instantiating our generic $\boldsymbol{\delta}$-rule as follows:

– Δ^s is the collection of all $\boldsymbol{\delta}$-formulae $\delta^s = (\exists x)QKey(\delta_0(x), x)$, where δ is a $\boldsymbol{\delta}$-formula of $\mathcal{L}^+_{\Sigma_{\mathbf{sko}}}$.

– For each $\delta^s \in \Delta^s$, $\mathbf{sko}_{[\delta^s]}$ is a set of function symbols new to δ^s, containing countably many symbols of arity equal to $|Free(\delta^s)|$, one for each quasi-key formula of $[\delta^s]$; additionally, we require that the sets $\mathbf{sko}_{[\delta^s]}$ are pairwise disjoint and form a partition of **sko**.
– Given a $\boldsymbol{\delta}$-formula δ occurring on a branch θ of a tableau $\mathcal{T}$, we put:
  - $\delta^a = \delta^s = \xi = (\exists x) QKey(\delta_0(x), x)$,
  - σ is the substitution such that $\delta = \delta^a \sigma$;
  - $\overrightarrow{H}$ is a tuple containing all the free variables in δ^s, ordered as they appear on δ^s from left to right, while f is the function symbol in $\mathbf{sko}_{[\delta^s]}$ associated to δ^s.

It can be easily checked that condition **C**0 is satisfied. Concerning conditions **C**1, **C**2, and **C**3, these are fulfilled by the definition of quasi-key formula. Condition **C**4 is satisfied since $\delta = \delta^a \sigma$, whereas conditions **C**5 and **C**6 are trivially satisfied since $\delta^a = \delta^s = \xi$. Finally, condition **C**7 is fulfilled, in view of the fact that $[\delta_0^a(f(\overrightarrow{H}))]\sigma = \delta_0(f(\overrightarrow{H})\sigma)$.

6 Isomorphism between tableau proofs with the $\boldsymbol{\delta}^{\mathbf{sk}}$-rule and with the $\boldsymbol{\delta}^{\epsilon}$-rule

Let us denote by $\mathcal{C}^{\mathbf{sk}}$ and $\mathcal{C}^{\epsilon}$ the tableau calculi obtained from the one defined in Section 2.4 by instantiating the generic $\boldsymbol{\delta}$-rule to the $\boldsymbol{\delta}^{\mathbf{sk}}$-rule, and by replacing the generic $\boldsymbol{\delta}$-rule with the $\boldsymbol{\delta}^{\epsilon}$-rule, respectively. Then the following result holds.

Theorem 2. *Let φ be a formula of $\mathcal{L}^+_{\Sigma}$. Then a tableau $\mathcal{T}^{\mathbf{sk}}$ for φ, relative to the calculus $\mathcal{C}^{\mathbf{sk}}$, is satisfied by a canonical model $\boldsymbol{\mathcal{M}}_{\mathbf{sko}}$ if and only if there exists a tableau $\mathcal{T}^{\epsilon}$ for φ, relative to the calculus $\mathcal{C}^{\epsilon}$, which is isomorphic to $\mathcal{T}^{\mathbf{sk}}$ and is satisfied by a pre-structure $\mathfrak{M}$ for $\mathcal{L}^+_{\Sigma_\epsilon}$.* □

The intuition behind such result is that the application of either the $\boldsymbol{\delta}^{\mathbf{sk}}$-rule or the $\boldsymbol{\delta}^{\epsilon}$-rule within a fixed, generic tableau calculus, gives rise to identical proofs up to the kind of terms (either Skolem terms or ϵ-terms) used to expand the existentially quantified formulae. Notice also that since the $\boldsymbol{\delta}^{\mathbf{sk}}$-rule is an instance of the generic $\boldsymbol{\delta}$-rule, the soundness of the $\boldsymbol{\delta}^{\epsilon}$-rule can be derived as a direct consequence.

For a proof of Theorem 2 the reader is referred to the extended version of the present paper [5]. Here, we limit ourselves to highlight its main ingredients, namely: (a) the construction of a bijection from $\mathcal{L}^+_{\Sigma_{\mathbf{sko}}}$ to $\mathcal{L}^+_{\Sigma_\epsilon}$, to map tableaux for $\mathcal{C}^{\mathbf{sk}}$ into isomorphic tableaux for $\mathcal{C}^{\epsilon}$, and vice versa; (b) a mechanism to construct a pre-structure $\mathfrak{M}$ for $\mathcal{L}^+_{\Sigma_\epsilon}$ (which we call *canonical*) out of a canonical model $\boldsymbol{\mathcal{M}}_{\mathbf{sko}}$ for $\mathcal{L}^+_{\Sigma_{\mathbf{sko}}}$ and vice versa, to show the equisatisfiability of isomorphic tableaux.

To such purpose, it is useful to stratify the language $\mathcal{L}^+_{\Sigma_\epsilon}$ and show also how to construct a pre-structure for $\mathcal{L}^+_{\Sigma_\epsilon}$, starting from a pre-structure for $\mathcal{L}^+_{\Sigma}$ (which, basically, can be identified with a standard classical structure).

The construction of canonical pre-structures is based on the adaptation of the concept of quasi-key formula to the formulae of $\mathcal{L}^+_{\Sigma_\epsilon}$. Further, the pre-structure resulting from such construction is substitutive; in fact, it can be proved that it satisfies the three conditions introduced in Definition 2.

The stratified language $\mathcal{L}^+_{\Sigma_\epsilon}$ Let $\mathcal{L}^+_{\Sigma}$ be the language of the sentence to be proved, not containing ϵ-terms.

Then we set $\mathcal{L}^+_{\Sigma_{\epsilon_0}} =_{Def} \mathcal{L}^+_{\Sigma}$ and, for $i \geq 1$, we put recursively

$$\mathcal{L}^+_{\Sigma_{\epsilon_{i+1}}} =_{Def} \mathcal{L}^+_{\Sigma_{\epsilon_i}} \cup \{\varphi : \varphi \text{ contains only } \epsilon\text{-terms } \epsilon x.\varphi' \text{ such that } \varphi' \in \mathcal{L}^+_{\Sigma_{\epsilon_i}}\}.$$

Finally, we put $\mathcal{L}^+_{\Sigma_\epsilon} =_{Def} \bigcup_{i=0}^{\infty} \mathcal{L}^+_{\Sigma_{\epsilon_i}}$.

Further, we define a notion of rank for formulae in $\mathcal{L}^+_{\Sigma_\epsilon}$ by putting

$$rank_\epsilon(\varphi) =_{Def} \min\{i \in \mathbb{N} : \varphi \text{ is in the language } \mathcal{L}^+_{\Sigma_{\epsilon_i}}\}.$$

The notion of quasi-key formula, defined in Section 5, can be easily applied to formulae of $\mathcal{L}^+_{\Sigma_\epsilon}$ as well.

Example 5. The quasi-key formula of $\varphi = P(x, h(z), \epsilon y.Q(z,y))$ with respect to the variable x is $\varphi_1 = P(\mathsf{x}_0, \mathsf{x}_1, \mathsf{x}_2)$, where $\sigma = \{\mathsf{x}_0 \leftarrow x, \mathsf{x}_1 \leftarrow h(z), \mathsf{x}_2 \leftarrow \epsilon y.Q(z,y)\}$ is a substitution such that $\varphi = \varphi_1\sigma$. □

Canonical pre-structures. A canonical pre-structure $\mathfrak{M} = \langle \boldsymbol{\mathcal{D}}, \boldsymbol{\mathcal{I}}, \boldsymbol{\epsilon\text{-}val} \rangle$ for the language $\mathcal{L}^+_{\Sigma_\epsilon}$ is defined from a first-order structure $\boldsymbol{\mathcal{M}} = \langle \boldsymbol{\mathcal{D}}, \boldsymbol{\mathcal{I}} \rangle$ for $\mathcal{L}^+_{\Sigma}$, by constructing the function $\boldsymbol{\epsilon\text{-}val}$ according to the stratification of the language as shown in below.

For every $\boldsymbol{\delta}$-formula δ of $\mathcal{L}^+_{\Sigma_\epsilon}$, let $[\mathcal{A}]_\delta$ be the equivalence class of all the assignments $\mathcal{A}_1, \mathcal{A}_2$ such that $\mathcal{A}_1 |_{Free(\delta)} = \mathcal{A}_2 |_{Free(\delta)}$. We introduce a family of functions $\{\boldsymbol{\epsilon\text{-}key}_i\}_{i\geq 1}$ to interpret all the quasi-key formulae of $\mathcal{L}^+_{\Sigma_\epsilon}$, in such a way that each $\boldsymbol{\epsilon\text{-}key}_i$ maps each pair $\langle \epsilon x.\delta_0, [\mathcal{A}]_\delta \rangle$, where $\epsilon x.\delta_0$ is the ϵ-term relative to a quasi-key formula δ_0 of rank $i-1$, to an element of $\boldsymbol{\mathcal{D}}$. Further, we define the family of functions $\{\boldsymbol{\epsilon\text{-}val}_i\}_{i\geq 0}$, to extend the interpretation to all the formulae of the language.

- For every $\boldsymbol{\delta}$-formula δ of $\mathcal{L}^+_{\Sigma_\epsilon}$ and assignment $\mathcal{A}$, we put

$$\boldsymbol{\epsilon\text{-}val}_0(\epsilon x.\delta_0, \mathcal{A}) =_{Def} \boldsymbol{d},$$

  where $\boldsymbol{d}$ is an arbitrary element of the domain $\boldsymbol{\mathcal{D}}$.

(1.) $P(x_1, x_2)$	(1.) $P(x_1, x_2)$
(2.) $(\forall z)(\exists x)\neg P(x, g(z))$	(2.) $(\forall z)(\exists x)\neg P(x, g(z))$
(3.) $(\exists x)\neg P(x, g(x_3))$	(3.) $(\exists x)\neg P(x, g(x_3))$
(4.) $\neg P(f(g(x_3)), g(x_3))$	(4.) $\neg P(\epsilon x.\neg P(x, g(x_3))), g(x_3))$

Fig. 2. Two isomorphic tableaux relative to the calculi $\mathcal{C}^{\mathbf{sk}}$ and $\mathcal{C}^{\epsilon}$, for the formula in Example 6.

- For every quasi-key formula δ_0 of rank i, with $i \geq 0$, if $\langle \boldsymbol{\mathcal{D}}, \boldsymbol{\mathcal{I}}, \boldsymbol{\epsilon\text{-}val}_i \rangle, \mathcal{A}' \models \delta$, for any $\mathcal{A}'$ in $[\mathcal{A}]_\delta$, we put

$$\boldsymbol{\epsilon\text{-}key}_{i+1}(\epsilon x.\delta_0, [\mathcal{A}]_\delta) =_{Def} \boldsymbol{c}\,,$$

  where $\boldsymbol{c}$ is an element of $\boldsymbol{\mathcal{D}}$ such that $\langle \boldsymbol{\mathcal{D}}, \boldsymbol{\mathcal{I}}, \boldsymbol{\epsilon\text{-}val}_i \rangle, \mathcal{A}'[x \leftarrow \boldsymbol{c}] \models \delta_0$, otherwise we put

$$\boldsymbol{\epsilon\text{-}key}_{i+1}(\epsilon x.\delta_0, [\mathcal{A}]_\delta) =_{Def} \boldsymbol{d}\,,$$

  where $\boldsymbol{d}$ is an arbitrary element of the domain.
- For every formula δ of $\mathcal{L}^+_{\Sigma_\epsilon}$ and assignment $\mathcal{A}$, we define $\boldsymbol{\epsilon\text{-}val}_{i+1}(\epsilon x.\delta_0, \mathcal{A})$ by distinguishing the following cases:
  - if $rank_\epsilon(\delta) < i$, then we put

$$\boldsymbol{\epsilon\text{-}val}_{i+1}(\epsilon x.\delta_0, \mathcal{A}) =_{Def} \boldsymbol{\epsilon\text{-}val}_i(\epsilon x.\delta_0, \mathcal{A})\,;$$

  - if $rank_\epsilon(\delta) = i$, φ is the quasi-key formula of δ, and σ is a substitution such that $\delta_0 = \varphi\sigma$, then we put

$$\boldsymbol{\epsilon\text{-}val}_{i+1}(\epsilon x.\delta_0, \mathcal{A}) =_{Def} \boldsymbol{\epsilon\text{-}key}_{rank_\epsilon(\varphi)+1}(\epsilon x.\varphi, [\mathcal{A}_1]_{(\exists x\varphi)})\,,$$

    where $\mathcal{A}_1 = \mathcal{A}[x_j \leftarrow (x_j\sigma)^{\boldsymbol{\mathcal{I}}, \boldsymbol{\epsilon\text{-}val}_i, \mathcal{A}}]_{x_j \in Free(\exists x\varphi)}$;
  - otherwise, namely when $rank_\epsilon(\delta) > i$, we put

$$\boldsymbol{\epsilon\text{-}val}_{i+1}(\epsilon x.\delta_0, \mathcal{A}) =_{Def} \boldsymbol{d}\,,$$

    where $\boldsymbol{d}$ is an arbitrary element of the domain.

Finally, we put $\boldsymbol{\epsilon\text{-}val}(\epsilon x.\delta_0, \mathcal{A}) =_{Def} \boldsymbol{\epsilon\text{-}val}_{rank_\epsilon(\delta)+1}(\epsilon x.\delta_0, \mathcal{A})$.

Example 6. Let us consider the unsatisfiable formula $\varphi = (\forall x)(\forall y)(P(x, y) \wedge (\forall z)(\exists x)\neg P(x, g(z)))$. In Figure 2 we describe two isomorphic tableaux for φ. The first one is relative to the calculus $\mathcal{C}^{\mathbf{sk}}$, the second to $\mathcal{C}^{\epsilon}$. For space reasons we have not reported the application of the $\boldsymbol{\alpha}$-rule and the first two applications of $\boldsymbol{\gamma}$-rule, resulting in the instantiation of the formula with the free variables x_1, x_2.

Both tableaux can be expanded to closed tableaux. The first one through the application of the substitution $\tau_1 = \{x_1 \leftarrow f(g(x_3)), x_2 \leftarrow g(x_3)\}$, the second one by means of $\tau_2 = \{x_1 \leftarrow \epsilon.x\neg P(x, g(x_3)), x_2 \leftarrow g(x_3)\}$.

The two tableaux are identical up to the terms used for the instantiation of the $\boldsymbol{\delta}$-formula. $(\exists x)\neg P(x, g(x_3))$ has as quasi-key formula $\varphi_1 = \neg P(\mathsf{x}_0, \mathsf{x}_1)$ and as corresponding substitution $\sigma = \{\mathsf{x}_0 \leftarrow x, \mathsf{x}_1 \leftarrow g(x_3)\}$. The Skolem term $f(g(x_3))$ used by the $\boldsymbol{\delta}^{\mathbf{sk}}$-rule in the first proof is equal to $f(\mathsf{x}_1)\sigma$, where f is the Skolem symbol associated to φ_1. The ϵ-term $\epsilon x.\neg P(x, g(x_3))$ used by the $\boldsymbol{\delta}^{\epsilon}$-rule in the second proof is equal to $\epsilon x.\neg P(x, \mathsf{x}_1)\sigma$, where $\epsilon x.\neg P(x, \mathsf{x}_1)$ is the ϵ-term associated to φ_1 when x_0 instantiates the quantified variable x. □

7 Conclusions

The result of isomorphism between tableaux adopting the $\boldsymbol{\delta}^{\epsilon}$-rule and tableaux applying the $\boldsymbol{\delta}^{\mathbf{sk}}$-rule witnesses that the Skolem terms constructed by the $\boldsymbol{\delta}^{\mathbf{sk}}$-rule and the ϵ-terms produced by the $\boldsymbol{\delta}^{\epsilon}$-rule can be considered as being essentially the same thing.

Such result is a point of conjuncture of two tendencies. The first one, relative to $\boldsymbol{\delta}$-rule variants based on Skolemization, relies on the construction of Skolem terms in a more natural way, reflecting the meaning of the instantiation. The second one, relative to the $\boldsymbol{\delta}^{\epsilon}$-rule, is based on the treatment of ϵ-terms as common terms of the language, where suitable constraints are imposed in the corresponding semantical structures.

The identification of Skolem terms and ϵ-terms has already been carried out in a purely proof theoretic way in [8], in the context of a free variable version of the predicate calculus, applicable in the field of automated deduction.

Our definition of the $\boldsymbol{\delta}^{\mathbf{sk}}$-rule (and, more generally, of our generic $\boldsymbol{\delta}$-rule) is also based on the constructions presented in [8], but keeps into account model-theoretic aspects particularly important in the context of semantic tableaux.

The two approaches to the elimination process of existential quantifiers in the context of semantic tableaux through the introduction of Skolem terms or ϵ-terms needs further investigation. In particular, we intend to compare variants of the $\boldsymbol{\delta}$-rule based on Skolemization with the $\boldsymbol{\delta}^{\epsilon}$-rule, endowed with extensional semantics (more appropriate for interactive theorem provers and calculi with equality). We also plan to further generalize the $\boldsymbol{\delta}$-rule framework introduced in [4], so as to embrace also $\boldsymbol{\delta}$-rules based on ϵ-terms.

References

1. Matthias Baaz and Christian G. Fermüller. Non-elementary speedups between different versions of tableaux. In Peter Baumgartner, Reiner Hähnle, and Joachim Posegga, editors, *Automated Rea-*

soning with Analytic Tableaux and Related Methods, International Conference, TABLEAUX'95, volume 918 of *Lecture Notes in Computer Science*, pages 217–230. Springer, 1995.
2. Bernhard Beckert, Reiner Hähnle, and Peter H. Schmitt. The even more liberalized delta-rule in free variable semantic tableaux. In Georg Gottlob, Alexander Leitsch, and Daniele Mundici, editors, *Third Kurt Gödel Colloquium*, volume 713 of *Lecture Notes in Computer Science*, pages 108–119. Springer, 1993.
3. Domenico Cantone and Marianna Nicolosi Asmundo. A further and effective liberalization of the *delta*-rule in free variable semantic tableaux. In Ricardo Caferra and Gernot Salzer, editors, *Automated Deduction in Classical and Non-Classical Logics*, volume 1761 of *Lecture Notes in Computer Science*, pages 109–125. Springer, 2000.
4. Domenico Cantone and Marianna Nicolosi Asmundo. A sound framework for δ-rule variants in free variable semantic tableaux. In Reinhold Letz, editor, *Proc. of the Fifth International Workshop on First-Order Theorem Proving (FTP 2005)*, Research Report 13/2005, pages 51–69. Research Report 13/2005, Universität Koblenz-Landau, Institut für Informatik, 2005. A revised version of the paper will appear in the special issue of the *Journal of Automated Reasoning* on *Automated Reasoning with Analytic Tableaux and Related Methods (TABLEAUX 2005).*
5. Domenico Cantone and Marianna Nicolosi Asmundo. *Skolem functions and Hilbert's ϵ-terms in free variable tableau systems (extended version).* Available at `http://www.dmi.unict.it/~cantone/papers/CNA06ext.pdf`, 2006.
6. Domenico Cantone, Marianna Nicolosi Asmundo, and Eugenio G. Omodeo. Global Skolemization with grouped quantifiers. In Moreno Falaschi, Marisa Navarro, and Alberto Policriti, editors, *Proc. of 1997 Joint Conf. on Declarative Programming, APPIA-GULP-PRODE'97*, pages 405–414, 1997.
7. Marcello D'Agostino, Dov M. Gabbay, Reiner Hähnle, and Joachim Posegga, editors. *Handbook of Tableau Methods*. Kluwer, Dordrecht, 1999.
8. Martin Davis and Ronald Fechter. A free variable version of the first-order predicate calculus. *J. Log. Comput.*, 1(4):431–451, 1991.
9. Melvin C. Fitting. A modal logic ε-calculus. *Notre Dame J. Formal Logic*, 16(1):1–16, 1975.
10. Melvin C. Fitting. *First-Order Logic and Automated Theorem Proving*. Graduate Texts in Computer Science. Springer-Verlag New York, Inc., 2nd edition, 1996.
11. Martin Giese and Wolfgang Ahrendt. Hilbert's ϵ-terms in automated theorem proving. In *Automated Reasoning with Analytic Tableaux and Related Methods, International Conference, TABLEAUX '99*, volume 1617 of *Lecture Notes in Computer Science*, pages 171–185. Springer, 1999.
12. Reiner Hähnle. Tableaux and related methods. In Robinson and Voronkov [21], chapter 3, pages 100–178.
13. Reiner Hähnle and Peter H. Schmitt. The liberalized delta-rule in free variable semantic tableaux. *J. Autom. Reasoning*, 13(2):211–221, 1994.
14. A. C. Leisenring. *Mathematical Logic and Hilbert's Epsilon-Symbol*. MacDonald, 1969.
15. Reinhold Letz. First-order tableau methods. In Robinson and Voronkov [21], pages 125–196.
16. Grigori Mints. Thoralf Skolem and the epsilon substitution method for predicate logic. *Nordic Journal of Philosophical Logic*, 1:133–146, 1996.
17. Georg Moser. The epsilon substitution method. Master's thesis, University of Leeds, 2000.
18. Tobias Nipkow, Lawrence C. Paulson, and Markus Wenzel. *Isabelle/HOL — A Proof Assistant for Higher-Order Logic*, volume 2283 of *LNCS*. Springer, 2002.
19. Andreas Nonnengart and Christoph Weidenbach. Computing small clause normal forms. In Robinson and Voronkov [21], pages 335–367.
20. Eugenio G. Omodeo, Domenico Cantone, Alberto Policriti, and Jacob T. Schwartz. A computerized Referee. To appear in Marco Schaerf and Oliviero Stock, editors, *Reasoning, Action and Interaction in AI Theories and Systems — Festschrift in Honor of Luigia Carlucci Aiello*, *Lecture Notes in Artificial Intelligence*, Springer, 2006.
21. John Alan Robinson and Andrei Voronkov, editors. *Handbook of Automated Reasoning (in 2 volumes)*. Elsevier and MIT Press, 2001.
22. Raymond M. Smullyan. *First-Order Logic*. Springer-Verlag, Berlin, 1968.
23. Klaus von Heusinger. Definite descriptions and choice functions. In Seiki Akama, editor, *Logic, Language and Computation*, pages 61–91. Kluwer, Dordrecht, 1997.