Tableaux per la Logica del Primo Ordine

Domenico Cantone

Dipartimento di Matematica e Informatica
Università di Catania

Logica Computazionale, A.A. 2006/07

Outline

1. **Syntax and Semantics**
   - Syntax
   - Semantics
2. **A tableau calculus for first-order logic**
   - The calculus
   - Soundness
   - Completeness
     - Hintikka sets
     - Tableaux construction rules and saturation strategies
   - Applications to the decision problem
   - Variants of the first-order tableau calculus
     - Tableaux with equality
     - Handling logical consequence
     - Free-variable tableaux

Outline

1 Syntax and Semantics
- Syntax
- Semantics

2 A tableau calculus for first-order logic
- The calculus
- Soundness
- Completeness
  - Hintikka sets
  - Tableaux construction rules and saturation strategies
- Applications to the decision problem
- Variants of the first-order tableau calculus
  - Tableaux with equality
  - Handling logical consequence
  - Free-variable tableaux

Let $\mathcal{L}_\Sigma$ be the first-order language over the signature

$$\Sigma = \{\, \mathcal{F}, \mathcal{P} \,\},$$

where

- $\mathcal{F}$ is a finite or countable set of *function symbols* of given finite arities, and
- $\mathcal{P}$ is a finite or countable set of *relation symbols*, or *predicate symbols*, of given finite arities.

Terms, *atomic formulas*, *literals*, and *formulas* of $\mathcal{L}_\Sigma$ are defined in the standard way.

Our main purpose is to present a *tableau calculus* which derives exactly the *valid sentences* of $\mathcal{L}_\Sigma$.

Following Smullyan's unifying notation, we divide the quantified sentences in γ- and δ-sentences, according to the table below.

A handy notation is also given for the *instances* of their matrices where there is a term t in place of the free variable.

In the following, it should be understood that by α, β, γ, and δ we denote an α-, β-, γ-, and δ-sentence, respectively.

γ	$\gamma(t)$
$(\forall x)\ \varphi$	φ^x_t
$\neg(\exists x)\ \varphi$	$\neg\varphi^x_t$

δ	$\delta(t)$
$(\exists x)\ \varphi$	φ^x_t
$\neg(\forall x)\ \varphi$	$\neg\varphi^x_t$

Table: γ- and δ-sentences

We also extend the signature $\mathbf{\Sigma}$ with a denumerable set **sko** of new constant symbols, called *parameters*, such that

$$\textbf{sko} \cap \mathcal{F} = \emptyset.$$

We put $\mathbf{\Sigma}^{\textbf{sko}} = \mathbf{\Sigma} \cup \textbf{sko}$ and we denote by $\mathcal{L}^{\textbf{sko}}$ the first-order language over the signature $\mathbf{\Sigma}^{\textbf{sko}}$.

First-order semantics can be defined in the standard way, by means of *models* $\mathbf{M} = \langle \mathbf{D}, \Im \rangle$ of $\mathcal{L}_{\Sigma}$, with $\mathbf{D}$ a nonempty *domain* and $\Im$ an *interpretation* of the symbols of $\mathcal{L}_{\Sigma}$, using also the usual notion of *assignments*.

Outline

1 Syntax and Semantics
- Syntax
- Semantics

2 A tableau calculus for first-order logic
- The calculus
- Soundness
- Completeness
  - Hintikka sets
  - Tableaux construction rules and saturation strategies
- Applications to the decision problem
- Variants of the first-order tableau calculus
  - Tableaux with equality
  - Handling logical consequence
  - Free-variable tableaux

The rules of the tableau calculus for the first-order language $\mathcal{L}_\Sigma$ are the following.

$$\frac{\alpha}{\begin{array}{c}\alpha_1\\ [\alpha_2]\end{array}}\ (\alpha\text{-rule}) \qquad \frac{\beta}{\beta_1 \mid \beta_2}\ (\beta\text{-rule})$$

$$\frac{\gamma}{\gamma(t)}\ (\gamma\text{-rule}) \qquad \frac{\delta}{\delta(p)}\ (\delta\text{-rule})$$

(t can be any ground term of $\mathcal{L}^{\text{sko}}$)

(p can be any new parameter in **sko** not already present on the branch to which the rule is applied)

Table: Tableau calculus for first-order logic

First-order tableaux for $\mathcal{L}_\Sigma$-sentences can be constructed much in the same way as propositional tableaux.
The expansion rule, though, must also consider the case in which the *nonliteral sentence* selected for expanding a branch ϑ is a γ- or a δ-sentence:

- in the case of a sentence γ, the branch ϑ can be prolonged with any instance $\gamma(t)$, for any ground term t of $\mathcal{L}^{\mathbf{sko}}$, whereas
- in the case of a sentence δ, the branch ϑ can be prolonged with any instance $\delta(p)$, where p is a parameter in **sko** *not* already occurring in ϑ.

Notice that symbols in **sko** are used as *temporary objects* and therefore are allowed in sentences labeling internal tableau nodes only.
The definitions of *closed/open branches*, *closed/open tableaux*, *tableau theorems*, and *tableau proofs* are the same as in the propositional case.

Soundness of the tableau calculus just outlined is based mainly on the properties below, which express the correctness of the γ- and δ-rules. We introduce first the following terminology, to state such properties conveniently.

Definition

Let $\mathfrak{I}$ be an interpretation of $\mathcal{L}^{\mathbf{sko}}$ and let $\xi \in \mathcal{P} \cup \mathcal{F} \cup \mathbf{sko}$. An interpretation $\mathfrak{I}'$ of $\mathcal{L}^{\mathbf{sko}}$ is a ξ*-variant* of $\mathfrak{I}$, if it coincides with $\mathfrak{I}$ on all symbols of $\mathcal{L}^{\mathbf{sko}}$ with ξ the only possible exception. □

Then we have:

Lemma (Exercise)

Let $\mathfrak{I}$ be an interpretation of $\mathcal{L}^{\mathbf{sko}}$.

(A) *If $\gamma^{\mathfrak{I}} = \mathbf{t}$, then $\gamma(\boldsymbol{t})^{\mathfrak{I}} = \mathbf{t}$, for every ground term $\boldsymbol{t}$ of $\mathcal{L}^{\mathbf{sko}}$.*

(B) *If $\delta^{\mathfrak{I}} = \mathbf{t}$, then for every parameter $\boldsymbol{p} \in \mathbf{sko}$ not occurring in δ there exists a $\boldsymbol{p}$-variant $\mathfrak{I}'$ of $\mathfrak{I}$ such that $\delta(\boldsymbol{p})^{\mathfrak{I}'} = \mathbf{t}$.*

Theorem (Soundness of first-order tableaux (Exercise))

Every theorem of our tableau calculus for first-order logic is a valid sentence.

Proof

Let φ be a sentence of $\mathcal{L}_\Sigma$ having a tableau proof and let **T** be a closed tableau for $\neg\varphi$.
If φ were not valid, then there would exist an interpretation $\mathfrak{I}$ such that $\varphi^{\mathfrak{I}} = \mathbf{f}$.
It can be shown by induction that **T** must have a branch ϑ whose sentences are made true by an interpretation $\mathfrak{I}'$ which differs from $\mathfrak{I}$ only in the evaluation of at most finitely many parameters in **sko**.
Hence ϑ cannot be closed, which is a contradiction. □

The *completeness proof* for first-order tableaux can be carried out along much the same lines as in the propositional case.

Particular care must be taken in the *tableau saturation process*, since, according to *Church's result on the undecidability of first-order logic*, saturation need not necessarily terminate.

Thus, it is an important issue to assure that selections of the branches and sentences to be expanded obey a *fairness criterion*.

The notion of *saturation strategies* is better formulated in terms of first-order *Hintikka sets*.

Definition (First-order Hintikka set)

Let **H** be a set of sentences of a first-order language $\mathcal{L}_{\Sigma}$ and let $\mathcal{C}_{\mathbf{H}}$ and $\mathcal{F}_{\mathbf{H}}$ be, respectively, the set of constants and of functors of positive degree occurring in the sentences of **H**. Then **H** is a *first-order Hintikka set (relative to $\mathcal{L}_{\Sigma}$)* provided that

(a) $\mathcal{C}_{\mathbf{H}}$ is non-null;

(b) $\mathbf{f} \notin \mathbf{H}$, $\neg\mathbf{t} \notin \mathbf{H}$, and **H** contains no complementary literals;

(c) if α is in **H**, then **H** contains all α-components of α;

(d) if β is in **H**, then **H** contains at least one β-component of β;

(e) if γ is in **H**, then **H** contains all instances $\gamma(t)$ of γ, for t ranging over the Herbrand universe $\mathbb{H}(\mathcal{C}_{\mathbf{H}}, \mathcal{F}_{\mathbf{H}})$;

(f) if δ is in **H**, then **H** contains at least one instance $\delta(t)$ of δ, with t a ground term in $\mathbb{H}(\mathcal{C}_{\mathbf{H}}, \mathcal{F}_{\mathbf{H}})$. □

What is particularly important about Hintikka sets is that they are always satisfiable, as the following lemma shows.

Lemma (Hintikka)

Let $\mathbf{H}$ *be a first-order Hintikka set relative to* $\mathcal{L}_{\Sigma}$. *Then* $\mathbf{H}$ *is satisfiable.*

Proof

Let $\mathcal{C}_{\mathbf{H}}$, $\mathcal{F}_{\mathbf{H}}$, and $\mathcal{P}_{\mathbf{H}}$ be respectively the set of constants, of functors of positive degree, and of relators occurring in the sentences of $\mathbf{H}$. We first define a *Herbrand interpretation* $\mathcal{H}$ of $\mathcal{L}_{\mathcal{P}_{\mathbf{H}};\, \mathcal{C}_{\mathbf{H}} \cup \mathcal{F}_{\mathbf{H}}}$ by putting:

- $\mathbf{D}^{\mathcal{H}} = \mathbb{H}(\mathcal{C}_{\mathbf{H}}, \mathcal{F}_{\mathbf{H}})$;
- $t^{\mathcal{H}} = t$, for every $t \in \mathbb{H}(\mathcal{C}_{\mathbf{H}}, \mathcal{F}_{\mathbf{H}})$;
- $Q(t_1, \ldots, t_n)^{\mathcal{H}} = \mathbf{t}$ iff $Q(t_1, \ldots, t_n) \in \mathbf{H}$, for every n-ary relator $Q \in \mathcal{P}_{\mathbf{H}}$ and $t_1, \ldots, t_n \in \mathbb{H}(\mathcal{C}_{\mathbf{H}}, \mathcal{F}_{\mathbf{H}})$.

./.

Proof (cntd.)

By structural induction, it can easily be shown that the interpretation $\mathcal{H}$ satisfies **H**. Plainly, any interpretation $\mathfrak{I}$ of the language $\mathcal{L}_\Sigma$, which extends $\mathcal{H}$, satisfies **H** too. □

Exercise

Provide the missing details of the proof of Hintikka's Lemma. □

Exercise

Let **H** *be a first-order Hintikka set relative to a first-order language* $\mathcal{L}$ *and let* φ *be a sentence of* $\mathcal{L}$.
Show that if $\varphi \in$ **H** *then* $\neg\varphi \notin$ **H**. □

We are now ready to introduce the abstract concept of saturation strategies for first-order tableaux.

Definition

A *construction rule for first-order tableaux* is a procedure $\mathcal{R}$ which for any given sentence φ allows us to construct a possibly infinite sequence $\mathbf{T}_0, \mathbf{T}_1, \mathbf{T}_2, \ldots$ of tableaux for φ, called *tableau $\mathcal{R}$-sequence for φ*, such that

- $\mathbf{T}_0$ is the initial tableau for φ, and
- $\mathbf{T}_{i+1}$ is obtained from $\mathbf{T}_i$ by a legal application of a tableau expansion rule, for each i less than the length of the $\mathcal{R}$-sequence.

A construction rule $\mathcal{S}$ is a *saturation strategy for first-order tableaux* such that for any sentence φ, if we denote by $\mathbf{T}$ the limit of the tableau $\mathcal{S}$-sequence for φ, then either $\mathbf{T}$ is a closed tableau or it has a branch ϑ whose sentences form a first-order Hintikka set. □

The existence of a saturation strategy $\mathcal{S}$ for first-order tableaux promptly entails the *completeness* of our first-order tableau calculus.

Indeed, let φ be a valid sentence of $\mathcal{L}_{\Sigma}$ and let $\mathbf{T}$ be the limit of the tableau $\mathcal{S}$-sequence for $\neg\varphi$.

If $\mathbf{T}$ were not a closed tableau, then it would have a branch ϑ whose sentences form a first-order Hintikka set $\mathbf{H}$.

Hence, by Hintikka's lemma, $\mathbf{H}$ would be satisfiable, and so would $\neg\varphi \in \mathbf{H}$, contradicting the validity of φ.

Below we outline the definition of a *saturation strategy* $\mathcal{S}$ *for first-order tableaux*.

We notice that for a construction rule $\mathcal{R}$ to be a saturation strategy, it must be the case that

$\mathcal{R}$ tries to build first-order Hintikka sets on all open branches, since it is not known in advance which branch, if any, will turn out to be satisfiable.

Thus, it is necessary that:

(i) $\mathcal{S}$ can select each open branch for expansion *infinitely often*;

(ii) each occurrence of a γ-sentence on a given branch is selected by $\mathcal{S}$ for expansion *infinitely often*.

Requirement *(i)* is achieved by keeping all *open* branches in a *queue* $\mathcal{Q}_{open}$ in such a way that

- branches to be expanded are selected (and popped out) from one side of $\mathcal{Q}_{open}$, whereas
- new branches are enqueued in $\mathcal{Q}_{open}$ from the other side.

Concerning *(ii)*, it is enough to maintain a *queue* $\mathcal{Q}_{\vartheta}$ for each open branch ϑ of the tableau.

The queue $\mathcal{Q}_{\vartheta}$ keeps all sentences in ϑ which still need to be expanded.

It is helpful to assume that we have fixed

- an enumeration σ of the parameters in **sko**
- an enumeration τ of the ground terms of the first-order language $\mathcal{L}^{\text{sko}}$.

Then, once a branch ϑ is selected,

- the sentence φ with *highest priority* in the queue $\mathcal{Q}_\vartheta$ is selected for expansion,
- φ is popped out from $\mathcal{Q}_\vartheta$
- the appropriate tableau rule is applied to φ.

In particular,

- if $\mathcal{S}$ selects a δ-sentence, then the δ-rule is applied to it with the first parameter in the enumeration σ which has not yet been used on ϑ;
- if $\mathcal{S}$ selects a γ-sentence γ, then we put

  $$\mathit{UnusedTerms}(\vartheta,\gamma) = \{t \in \mathbb{H}(\mathcal{C}_\vartheta, \mathcal{F}_\vartheta) \;:\; \gamma(t) \text{ does not occur in } \vartheta\},$$

  where $\mathcal{C}_\vartheta$ and $\mathcal{F}_\vartheta$ are respectively the *constant symbols* and the *functors of positive arity* in ϑ, and we distinguish the following three cases:
  - if $\mathbf{UnusedTerms}(\vartheta,\gamma) \neq \emptyset$, then the γ-rule is applied to γ with the first term $t \in \mathbf{UnusedTerms}(\vartheta,\gamma)$ in the enumeration τ; or else
  - if $\mathcal{C}_\vartheta = \emptyset$ then the γ-rule is applied to γ with the first unused parameter in the enumeration σ; or else
  - no application of the γ-rule takes place at this step.

In all cases, the sentences which are added to the branch ϑ as a result of the application of one of the tableau rules are entered in the queue associated with ϑ (or with the new sibling of ϑ, in case ϑ has been split out by the β-rule).

In addition, if the selected sentence is a γ-sentence, it is *re-entered* in the queue:

> this is the reason for possible nontermination of the tableau construction process.

Termination condition

The tableau construction stops when either

- all branches of the tableau are closed, or
- the tableau has an open branch which cannot be further expanded by the rule $\mathcal{S}$.

In the former case, the initial sentence is clearly unsatisfiable, whereas in the latter case or in the case of nontermination it can be shown that the initial sentence is satisfiable.

Exercise

Verify that the construction rule $\mathcal{S}$ outlined above is a saturation strategy for first-order tableaux. □

From the previous discussion, we can immediately conclude:

Theorem (Completeness of first-order tableaux)

Our tableau calculus for first-order logic is complete. □

- According to the above termination condition for the saturation strategy $\mathcal{S}$, in some favorable cases the saturation process can *terminate* even for satisfiable sentences.
- It is therefore interesting to single out those collections of sentences for which $\mathcal{S}$ can work as a decision procedure.

A first condition which is plainly sufficient for termination is that during the tableau construction by $\mathcal{S}$, no γ-sentence is ever generated (condition C_0).

Another more interesting case arises when γ-sentences can be generated during the tableau construction. In such a case, termination is enforced provided that

- no *functor of positive degree* occurs in the initial sentence; and
- there exists an upper bound $k \in \mathbb{N}$ such that $|\mathcal{C}_\vartheta| \leq k$ holds, for each branch ϑ, during the tableau construction by $\mathcal{S}$.
  (Condition C_1)

Let us put:

- $\mathbf{L_0}$: collection of sentences of $\mathcal{L}_\Sigma$ for which condition C_0 holds;
- $\mathbf{L_1}$: collection of sentences of $\mathcal{L}_\Sigma$ for which condition C_1 holds.

Then, we plainly have that $\mathcal{S}$ is a decision procedure for the satisfiability problem for $\mathbf{L_0}$ and $\mathbf{L_1}$.

The preceding results, though, are quite unsatisfactory, since we do not have a syntactic characterization of $\mathbf{L_0}$ and $\mathbf{L_1}$.

We content ourselves with the following syntactic characterizations of *proper* subsets of $\mathbf{L_0}$ and $\mathbf{L_1}$.

Let us first define the concept of *essentially universal* and *essentially existential* subformulas.
For the sake of simplicity, we will assume that the propositional connectives of our first-order language $\mathcal{L}_\Sigma$ are just $\neg$, $\wedge$, and $\vee$, though the definition below can easily be adapted to the more general cases in which other connectives are present.

Definition

Let φ be a sentence of $\mathcal{L}_\Sigma$. An occurrence of a subformula of φ is *positive* if it falls within the scope of an *even* number of the negation symbol $\neg$, otherwise it is *negative*.

An occurrence of a quantified subformula of φ is *essentially existential* if either

- it occurs positively in φ and has the form $(\exists x)\psi$, or
- it occurs negatively in φ and has the form $(\forall x)\psi$.

An occurrence of a quantified subformula of φ which is not essentially existential is *essentially universal*. □

Let

- $\mathbf{L}'_0$: collection of sentences of $\mathcal{L}_\Sigma$ with no occurrence of essentially existential subformulas

  Example: $(\forall x)P(x) \wedge \neg(\exists x)R(x)$

- $\mathbf{L}'_1$: collection of sentences φ of $\mathcal{L}_\Sigma$ such that
  - φ involves no functor of positive degree;
  - no essentially universal quantifier falls within the scope of an essentially existential quantifier in φ.

  Example: $(\forall x_1)(\forall x_2)(\exists x_3)(\exists x_4)R(x_1, x_2, x_3, x_4)$

Then we have $\mathbf{L}'_0 \subset \mathbf{L}_0$ and $\mathbf{L}'_1 \subset \mathbf{L}_1$.

Exercise

Show that $\mathcal{S}$ is a decision procedure for the validity problem for $\mathbf{L}'_0$ and $\mathbf{L}'_1$.
Also prove the inclusions $\mathbf{L}'_0 \subset \mathbf{L}_0$ and $\mathbf{L}'_1 \subset \mathbf{L}_1$. □

Example ($\forall^*$-sentences)

- $\mathbf{L}_0''$: collection of sentences of $\mathcal{L}_\Sigma$ of the form

$$(\forall x_1)(\forall x_2)\cdots(\forall x_n)\psi$$

where ψ does not contain any quantifier.

Then $\mathbf{L}_0'' \subset \mathbf{L}_0'$. □

Example ($\forall^*\exists^*$-sentences)

- $\mathbf{L}_1''$: collection of sentences of $\mathcal{L}_\Sigma$ of the form

$$(\forall x_1)(\forall x_2)\cdots(\forall x_n)(\exists y_1)(\exists y_2)\cdots(\exists y_m)\psi$$

where ψ is unquantified and does involve any functor of positive arity.

Then $\mathbf{L}_1'' \subset \mathbf{L}_1'$. □

For the sake of simplicity, in the preceding section we limited ourselves to the discussion of an elementary version of a tableau calculus for first-order logic.

Among the several variants existing in the literature, we will briefly sketch three of them:

- tableaux with *equality*,
- tableaux for handling *logical consequences*, and
- tableaux with *free variables*.

Though such variants are presented separately, they could be merged into a *single tableau calculus*.

To handle *equality*, our tableau calculus can be supplemented with the two rules below, where t and u stand for ground terms of $\mathcal{L}^{\text{sko}}$.

$$\frac{}{t = t}\ (\textit{reflexivity rule}) \qquad \frac{\begin{array}{c} t = u \\ \varphi \end{array}}{\varphi^t_u}\ (\textit{replacement rule})$$

Table: Tableau equality rules

Notice that the reflexivity rule has no premiss. This means that at any time during the tableau construction it is possible to add an equality of type $t = t$.

To show whether a first-order sentence φ is a *logical consequence* of a set of sentences $\mathbf{\Phi}$, one can build a closed tableau for $\neg\varphi$ by means of the extension of our tableau calculus with the following $\mathbf{\Phi}$-introduction rule:

at any time during the tableau construction it is possible to add any sentence in $\mathbf{\Phi}$.

Our tableau calculus is not very well suited for automation, since in most cases when the γ-rule is applied on a branch, one does not have enough information for choosing the right *instance*, namely an instance which will contribute to an earlier closure of the branch.

A solution to this problem consists in allowing *free variables* in γ-instances, which later, when enough information becomes available, can be opportunely *instantiated* to close a branch by means of a *unification algorithm*.

Also, to cope with the presence of free variables, a different variant of the δ-rule must be used.

$$\frac{\gamma}{\gamma(x)} \ (\gamma\text{-rule})$$

(x is a free variable)

$$\frac{\delta}{\delta(f(x_1, \ldots, x_n))} \ (\delta\text{-rule})$$

($x_1, \ldots, x_n$ are all the free variables of δ and f is a new n-ary functor)