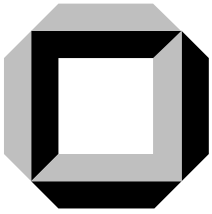
A-Ordered Tableaux

Reiner Hähnle
Stefan Klingenbeck

Interner Bericht 26/95



Universität Karlsruhe
Fakultät für Informatik

A-Ordered Tableaux

Reiner Hähnle and Stefan Klingenbeck

University of Karlsruhe
Institute for Logic, Complexity and Deduction Systems
76128 Karlsruhe, Germany
{reiner,klingenb}@ira.uka.de +49-721-608-{4329,3978}

Abstract

In resolution proof procedures refinements based on A-orderings of literals have a long tradition and are well investigated. In tableau proof procedures such refinements were only recently introduced by the authors of the present paper. In this paper we prove the following results: we give a completeness proof of A-ordered ground clause tableaux which is a lot easier to follow than the previous one. The technique used in the proof is extended to the non-clausal case as well as to the non-ground case and we introduce an ordered version of Hintikka sets that shares the model existence property of standard Hintikks sets. We show that A-ordered tableaux are a proof confluent refinement of tableaux and that A-ordered tableaux together with the connection refinement yield an incomplete proof procedure. We introduce A-ordered first-order NNF tableaux, prove their completeness, and we briefly discuss implementation issues.

1 Introduction

In resolution proof procedures refinements based on A-orderings[1] of literals have a long tradition and are well investigated. In tableau proof procedures such refinements were only recently introduced by the authors of the present paper [7]. The motivation for considering A-ordered tableaux is that in recent years tableau systems were increasingly used as proof procedure for applications in program verification. The verification of programs frequently requires proof plans or human interaction for difficult proof obligations and the analysis of failed proof attempts. Tableaux procedures support these tasks, because they do not need to transform proof obligations in clausal form and often distinguish cases in their branching behavior like human beings do. Special purpose provers for theories can be easily integrated in the tree structure of tableaux. A-ordered tableaux represent a refinement that is compatible with these goals. A-orderings restrict the search space and put one in a stronger position with respect to termination of non-theorems.

In this paper we prove the following results: in Section 2 we give a completeness proof of A-ordered ground clause tableaux which is a lot easier to follow than the one in [7]. The technique used in the proof has several more advantages: first, it can be extended to the non-clausal case as well as to the non-ground case—this is done in Section 3, where we intro-

[1] See Section 2 for a precise definition.

duce ordered links in general NNF formulas and in Section 4, where we introduce an ordered version of Hintikka sets that shares the model existence property of standard Hintikks sets. Second, from the proof it is immediate that A-ordered tableaux are a proof confluent [8] refinement. This property is of great importance for finding counter examples to non-theorems. In the light of this property it is not surprising that A-ordered tableaux together with the connection refinement [9] yield an incomplete proof procedure. Thus there is no hope of using A-orderings within such proof procedures as the connection method or model elimination all of which employ the connection refinement. This is also proved in Section 2. In Section 5 we define A-ordered first-order NNF tableaux, prove their completeness, and we briefly discuss implementation issues.

This paper provides an answer to the basic theoretical questions that arise from order-restricted tableaux. Implementation issues, computational results, and secondary theoretical issues such as the extension to equality and decidability results will be the topic of future papers.

2 Ordered Ground Clause Tableaux

Tableaux are defined as possibly infinite trees labelled with formulas. We use the terms node, root, leaf, and immediate successor without further explanation. A branch is either a finite path from the root to a leaf or an infinite path starting at the root. We denote the set of nodes on a path from the root to a node u by $pred(u)$. These nodes are called predecessors of u. If T is a tree whose nodes are labeled with literals and u is a node of T, we write $clause(u)$ for the set of literals labelling the immediate successors of u.

Definition 1 A **ground clause tableau** T for a set of ground clauses M is a tree for which the following holds:

1. Each node of T is labelled with a literal.
2. The root of T is labelled with the atom true.
3. For each node u that is not a leaf clause(u) appears in M.
4. For any distinct nodes $u \neq v$ on any branch clause$(u) \neq$ clause(v).

Let B be a branch of a ground clause tableau. A literal L is **on** B, if one of its nodes is labelled with L. A clause C is **on** B, if $C =$ clause(u) for some node $u \in B$.

Clause (4) in the previous definition is also known as *regularity* (cf. [9]) and it is a straightforward, but useful optimization which excludes unnecessarily long tableaux.

Definition 2 A tableau branch is **closed** if it contains a complementary pair of literals. A tableau is **closed** if all of its branches are closed. The tableau is called **open** otherwise.

Definition 3 An **A-ordering** on a set of atoms B is a binary relation $<_A$ such that for all $a, b, c \in B$

Irreflexivity $a \not<_A a$.

Transitivity $a <_A b$ and $b <_A c$ imply $a <_A c$.

Substitutivity $a <_A b$ implies $a\sigma <_A b\sigma$ for all substitutions σ.

Thus in the ground case an A-ordering simply is an irreflexive, transitive ordering.

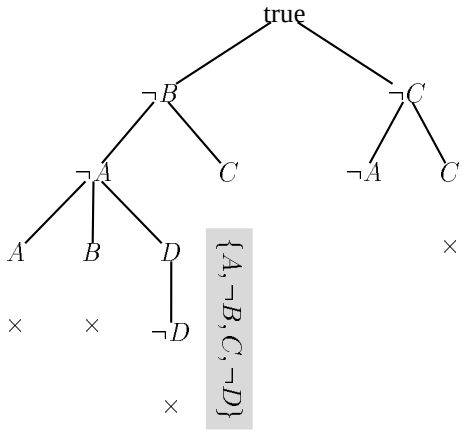
We assume in the following that $<$ is an A-ordering on the atoms of an arbitrary, but fixed signature.

Definition 4 We define an **A-ordered ground clause tableau**[2] for a ground clause set M as a ground clause tableau for M obeying the following extension rule restriction: for each node u that is not a leaf clause(u) contains a $<$-maximal literal which is (i) either complementary to a $<$-maximal literal occurring in another clause from M or (ii) is complementary to a literal in pred(u).

Example 1 Consider the A-ordering $D < B < A < C$ and the ground clause set

$$S{=}\{\neg B \vee \boxed{\neg C}, \boxed{A} \vee B \vee D, \neg A \vee \boxed{C}, \boxed{\neg D}\}.$$

The maximal $<$-literal in each clause is highlighted. A partial ordered ground clause tableau for S is as follows:



As the initial tableau is empty, only one of two clauses that contain complementary, maximal literals are allowed for the first step. In the present example, these are the first and third clause containing C and $\neg C$, respectively. We choose to expand with $\{\neg B \vee \neg C\}$ first. As B does not occur maximally in any clause the left branch can only be expanded using $\{\neg A \vee C\}$. Similarly, the same clause must be used to extend the right branch. Hence, the tableau up to this point is determined up to the order in which the first two clauses are being used.

Clauses one and three are already on the now leftmost branch which can only be extended with the clause containing A and then with the unit clause. This yields the tableau shown above. Note that it is not possible to extend the leftmost open branch even though there are clauses being not on it. Neither of them, however, may be used, because their maximal literals do not occur complemented on the branch. Indeed, it is possible to extend the literal set on the leftmost open branch to a model of S as indicated.

[2] In [7] we used a slightly different definition which is, however, easily seen to be equivalent to the present one.

Definition 5 Let M be a ground clause set and T an A-ordered ground clause tableau for M. T is called **saturated**, if there is no A-ordered ground clause tableau T' for M such that T is a proper subtree of T'.

Theorem 1 Let M be a (not necessarily finite) set of ground clauses and B an open branch of an A-ordered saturated ground clause tableau T for M. Then M has a model.

Proof In the following we identify as usual consistent sets of ground literals from M with (partial) interpretations of M.

In particular, since B is open, the set I of literals on B is consistent and, therefore, constitutes a partial interpretation of M. Consider the set of clauses M' that are not made true by I, that is, no literal of M' occurs in I. Let J be the set of literals occuring $<$-maximally in any clause of M'.

J gives rise to a well-defined partial interpretation, because no literal and its complement can occur maximally in clauses of M'; otherwise these clauses would be on B and not in M' by Definition 4, item (i) and Definition 5.

Finally, the interpretation $I \cup J$ is well-defined, for if L occurs maximally in a clause of M' (and hence, in J), then $\overline{L}$ cannot be in I by Definition 4, item (ii) and Definition 5.

By construction, the interpretation $I \cup J$ satisfies at least one literal in each clause of M, therefore, it satisfies M.

The argument of Theorem 1 provides a simple, effective procedure to calculate a model when M is finite. For instance, the model used in Example 1 was constructed this way.

Obviously, each A-ordered ground clause tableaux can be extended to a saturated A-ordered ground clause tableaux. Therefore, we can conclude that ordered ground clause tableaux are *proof confluent*, that is, each ordered ground clause tableau for an unsatisfiable formula is a subtree of a suitable closed ordered ground clause tableau. This observation has two important implications:

1. It is not necessary to backtrack over alternative selections of clauses to be used for extension.
2. In the ground case counter examples (models) for satisfiable formulas can be extracted directly from saturated branches via the construction given in the proof.

On the other hand, typical non-proof confluent tableau refinements are incompatible with the ordering restriction.

Definition 6 A clause tableau is said to obey the **connection condition** if every non-leaf node u but the root node must have a leaf node v as its direct successor such that the labels of u and v are complementary literals.

Theorem 2 Ordered ground clause tableaux and ground clause tableaux obeying the connection condition are incompatible, in other words, there is an unsatisfiable ground clause set M and an A-ordering $<$ such that there exists no $<$-ordered ground clause tableau for M which at the same time obeys the connection condition.

Proof Consider the ground clause set

$$M = \{A \vee \neg B,\ A \vee B \vee \neg C,\ C \vee D,\ C \vee \neg D,\ \neg A \vee \neg C \vee E,\ \neg E \vee F,\ \neg F\}.$$

Assume that $A < B < C < D < E < F$. If we start with either $C \vee D$ or $C \vee \neg D$, then, because of the connection condition, the left branch must be extended with $A \vee B \vee \neg C$ ($\neg A \vee \neg C \vee E$ is not admissible, because in it $\neg C$ is not maximal), but then it is impossible to extend the branch containing A. If we start with $\neg E \vee F$ or $\neg F$, then we arrive at an open branch containing $\neg E$ as a leaf. This can only be extended with $\neg A \vee \neg C \vee E$, because of the connection condition and we are stuck in the branch that contains $\neg A$. It is easy to construct a closed ordered tableau for M *without* the connection condition.

3 Ordered Links in NNF Formulas

When we extend A-ordered clause tableaux to arbitrary (ground) formulas in NNF the crucial question is: which sets of literal occurrences are to be ordered? In a formula such as $A \vee (B \wedge C)$, for instance, there are two *implicit* disjunctive clauses: $\{A, B\}$ and $\{A, C\}$. Such implicit clauses are usually called *paths* (more precisely: disjunctive paths) through a formula. In the case of a CNF formula the set of its clauses is identical to the set of its disjunctive paths. Appropriate tools for a formal treatment of the notions required here were developed by Andrews [1], Bibel [5] and Murray & Rosenthal [10]. Here we use a notation which is close to that of the latter paper (but this can be mainly considered as a matter of taste—the other formalisms could be used just as well).

Definition 7 An **NNF formula** is defined recursively as follows:

1. A (possibly non-ground) literal is an NNF formula.
2. If $\phi_1, \ldots, \phi_n$ ($n \geq 2$) are NNF formulas, then $\phi_1 \wedge \cdots \wedge \phi_n$ is an NNF formula as well. Each pair ϕ_i, ϕ_j ($i \neq j$) is said to be **conjoint** and the formula is called conjunctive.
3. If $\phi_1, \ldots, \phi_n$ ($n \geq 2$) are NNF formulas, then $\phi_1 \vee \cdots \vee \phi_n$ is an NNF formula as well. Each pair ϕ_i, ϕ_j ($i \neq j$) is said to be **disjoint** and the formula is called disjunctive.
4. If ϕ is an NNF formula and x does not occur bound in ϕ, then $(\forall x)\phi$ is an NNF formula as well.

Let ϕ be a closed NNF formula where different quantifiers bind different variables. Then we call the NNF formula that results when all quantifiers in ϕ are deleted the **matrix** of ϕ.

Let ψ be a closed NNF formula and ϕ its matrix. A **ground instance** of ψ is a ground NNF formula $\phi\tau$ in which the substitution τ replaces each variable of ϕ by a ground term.

Two subformulas A, B in an NNF formula ϕ are **c-connected (d-connected)** iff there are subformulas F, G in ϕ such that A is a subformula of F, B is a subformula of G, and F, G are conjoint (disjoint) in ϕ.

A **c-path (d-path)** through ϕ is a maximal set of pairwise c-connected (d-connected) literals in ϕ.

Sets of formulas are considered conjoint, which extends the definition of c-paths (d-paths) to sets of NNF formulas.

The following lemma is an immediate consequence of the definition of A-orderings.

Lemma 1 Let C be a clause or a d-path, L a literal of C and μ a substitution. If $L\mu$ is $<$-maximal in $C\mu$, then L is $<$-maximal in C.

Note that the statement of the previous lemma does not hold in the other direction. Consider, for example, the clause $C = \{p(x) \vee p(y)\}$. Since its literals are unifiable, they cannot be ordered by any A-ordering, hence both literals are maximal in C. On the other hand using the substitution $\sigma = \{x \leftarrow a, y \leftarrow b\}$ and an A-ordering which is a lexicographic ordering on ground terms we see that $p(x)\sigma$ is not maximal in $C\sigma$.

In the completeness proof for A-ordered ground clause tableau we used that a clause is satisfied by an interpretation if one of the clause's literals is satisfied by the interpretation. This result can be generalized to NNF formulas.

Proposition 1 (see, e.g., [10]) A ground NNF formula ϕ is satisfied by an interpretation I iff at least one literal in every d-path of ϕ is satisfied by I.

The basis of each refutation procedure is the detection of complementary pairs of literal occurrences. A pair of literal occurrences that might become complementary after a suitable instantiation is usually called a link. We employ this concept to adequately deal with NNF formulas.

Definition 8 Let Φ be a set of formulas in NNF and let τ, τ' be substitutions renaming all bound variables of Φ into new variables. A **link** in Φ is a pair (F, G) of c-connected literal occurrences in Φ such that $\{F\tau, \overline{G}\tau'\}$ is unifiable.

A unifier of $\{F\tau, \overline{G}\tau'\}$ is called **link-unifier** of (F, G). Let Φ be a set of formulas and ϕ a formula. We say ϕ **contains a link into** Φ, if there is a link (F, G) in $\Phi \cup \{\phi\}$ such that F occurs in ϕ and G in a formula of Φ.

Definition 9 Let ϕ be a formula or a set of formulas in NNF and let $<$ be an A-ordering on the atom set of ϕ. We say that a literal F **occurs $<$-maximally** in ϕ iff there is a d-path p through ϕ in which F occurs $<$-maximally. An **$<$-ordered link** in ϕ is a link (F, G) in ϕ such that both F and G occur maximally in ϕ.

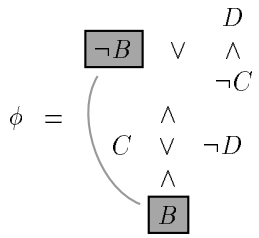
Note that if ϕ is of the form $\psi \wedge L$, where L is a literal, it is sufficient for the existence of an ordered link in ϕ that $\overline{L}$ occurs maximally in a d-path of ψ.

The basic idea of the ordered NNF tableau procedure will be to employ exactly the same restriction on universal quantifier rules as on the other formula expansion rules:

Expand $(\forall x)\phi(x)$ iff there is a formula ρ on the same branch such that $(\forall x)\phi(x) \wedge \rho$ contains an ordered link, in other words, it contains a link into the current branch. Moreover, any instance of $(\forall x)\phi(x)$ used for an extension must contain an ordered link into the current branch.

A subtle point which, however, occurs independently of using orderings or not is the case $\rho = (\forall x)\phi(x)$. Bound variables must be considered as pairwise different in the definition of an (ordered) link. This is exemplified with the formula $((\forall x)p(x) \vee p(f(x))) \wedge \neg p(x))$. Given the ordering $p(x) < p(f(x))$, there is exactly one ordered link (up to renaming) between this formula and a copy of itself, namely $(p(f(x)), \neg p(x))$. The atoms of the link are not unifiable, but as x occurs free in $\neg p(x)$ and it occurs bound in $((\forall x)p(x) \vee p(f(x)))$, the bound occurrence can be renamed appropriately, for instance, it can be renamed into $(p(f(x_1)), \neg p(x))$. Note that in the case when only the formula $(p(x) \vee p(f(x))) \wedge \neg p(x)$ occurs on a tableau branch, there is *no* ordered link to itself. These considerations are reflected in the previous definition of an ordered link in NNF formulas.

Example 2 Let $<$ be the downward lexicographic ordering on ground atoms: $B > C > D$. Consider the formula[3]



The d-paths of ϕ are $\{\{\neg B, D\}, \{\neg B, \neg C\}, \{C, \neg D\}, \{B\}\}$.

Some of the c-paths of ϕ are $\{\{\neg B, C, B\}, \{D, \neg C, C, B\}\}$ etc.

The following are the links of ϕ: $(\neg B, B)$, $(\neg C, C)$, $(D, \neg D)$. Only the first is an $<$-ordered link; it is shown explicitly in the graphical formula representation above.

4 Ordered Hintikka Sets

If we were only interested in first-order tableaux for CNF formulas it would be sufficient to lift ordered ground clause tableaux to first-order (which can be done straightforwardly) and use Theorem 1. For the NNF case, just as in classical logic, a little more work is required, because one needs to establish a non-clausal version of Herbrand's Theorem which preserves the structure of a tableau branch. In the present section we define the notion of an ordered Hintikka set for which the usual model existence theorem can be established. Its proof is a combination of the classical argument with the idea used in the proof of Theorem 1.

Definition 10 Let H, M be sets of closed NNF formulas. H is called **ordered Hintikka set** for M, if the following conditions hold:

1. $M \subset H$.
2. If $\phi_1 \wedge \cdots \wedge \phi_n \in H$ contains an ordered link into H, then $\{\phi_1, \cdots, \phi_n\} \subset H$.
3. If $\phi_1 \vee \cdots \vee \phi_n \in H$ contains an ordered link into H, then at least one of $\phi_1, \cdots, \phi_n$ is in H.
4. If for $(\forall x)\phi(x) \in H$ and a ground term t the formula $\phi(t)$ contains an ordered link into H, then $\phi(t) \in H$.
5. No subset $L \subseteq H$ that consists only of literals contains a link.

Note that all literals contained in a Hintikka set are ground literals.

Theorem 3 (Hintikka's Lemma, ordered version) Every ordered Hintikka set has a model.

[3] In order to ease readability as in [10] we use a two-dimensional notation for NNF formulas in which conjuncts are drawn vertically and disjuncts are drawn horizontally.

Proof We provide an interpretation and show by induction on the depth d of nesting of logical operators that this interpretation satisfies all formulas of H. Literals have a depth of 1 by definition. First we define a suitable interpretation for them.

The literals $\mathbf{L}$ of H define a partial interpretation I via $I = \mathbf{L}$. By clause (5) of the definition of an ordered Hintikka set, H does not contain complementary literals, hence I is well-defined.

Consider the set of formulas

$$\begin{aligned}\text{Unlinked} \quad = \quad & \{\psi_1 \vee \cdots \vee \psi_n \in B \mid \psi_i \notin B, \text{ for all } i = 1, \ldots, n\} \cup \\ & \{\psi_1 \wedge \cdots \wedge \psi_n \in B \mid \psi_i \notin B, \text{ for some } i = 1, \ldots, n\} \cup \\ & \{\phi(t) \mid (\forall x)\phi(x) \in H,\ \phi(t) \notin H,\ t \text{ ground}\}\end{aligned}$$

and the set J of literals which occur maximally in the set of ground instances of Unlinked.

First we show that J is a well-defined partial interpretation of the formulas on H. Assume that to some atom P different truth values were assigned in the definition of J. Then there must be ground instances of formulas $\psi, \rho \in$ Unlinked in which P and $\neg P$ occur maximally (ψ and ρ may be identical). By Lemma 1, literals that occur maximally in ground instances of a formula are ground instances of literals that occur maximally in the original formula. Hence, ψ and ρ contain a pair of literals (F, G) which has $(P, \neg P)$ as an instance. Therefore, F and $\overline{G}$ are unifiable modulo renaming of bound variables. Thus ψ (and ρ) contains an ordered link into H and, therefore, by one of clauses (2)–(4) of the definition of an ordered Hintikka set, cannot be in "Unlinked" which is a contradiction.

It remains to show that $I \cup J$ is still a well-defined partial interpretation of the formulas of H. Assume that to the same ground atom P were assigned different truth values in I and J, w.l.o.g. let $P \in I$ and $\neg P \in J$. Then, by definition of J, $\neg P$ occurs maximally in a ground instance of a formula $\psi \in$ Unlinked. By definition of I, H contains the literal P. Trivially, P occurs maximally in P. As before, we see that ψ contains an ordered link into H (via P), which contradicts $\psi \in$ Unlinked.

Thus $I \cup J$ is a well-defined partial interpretation of H. Moreover, it is a model of all ground instances of "Unlinked" by definition of J and by Proposition 1. By Herbrand's Theorem $I \cup J$ is as well a model of "Unlinked".

In the clausal case $I \cup J$ clearly constitutes already a model of H. In the NNF case we show by induction on the depth d of a formula (that is, the number of recursion steps needed in Definition 7 for its construction) in H that $I \cup J$ models H.

$d = 1$**:**

By definition, I satisfies the literals of H.

$d > 1$**:**

- If $\phi = \psi_1 \wedge \cdots \wedge \psi_n \in$ Unlinked, then ϕ is satisfied by J. If $\phi \notin$ Unlinked, then we apply the induction hypothesis to $\psi_1, \ldots, \psi_n$ to see that all of them are satisfied by $I \cup J$. Then, by the usual completeness lemma (see, for example, [6]) of the conjunction connective, ϕ is satisfied by $I \cup J$.

- Disjunctive formulas are treated analogously.

- All ground instances $\phi(t)$ of $(\forall x)\phi(x) \in H$ are contained either in H or in "Unlinked". Therefore, all $\phi(t)$ are satisfied by $I \cup J$ either by the induction hypothesis or by definition of J. This guarantees that $I \cup J$ models $(\forall x)\phi(x)$, this time by the usual completeness lemma for universally quantified formulas.

5 Ordered First-Order NNF Tableaux

5.1 Ordered first-order NNF tableau procedure

Now we are in a position to define an ordered first-order NNF tableau procedure. As usual we have rules for conjunction, disjunction and universally quantified formulas (as the input is in skolemized NNF we do not need rules for negated and existentially quantified formulas).

Definition 11 Let ϕ be a closed NNF formula. An **ordered NNF tableau** for ϕ is a finitary labelled tree constructed as follows:

Init The tree with a single node labelled with ϕ is an ordered NNF tableau for ϕ.

Con Assume T is already an ordered NNF tableau for ϕ, B is a branch of T, $\psi{=}\psi_1 \wedge \cdots \wedge \psi_n$ is on B, ψ has an ordered link into B and it had not yet a rule applied to it on B. Then B is extended by n new nodes each of which is labelled with one of the ψ_i. The resulting tableau again is an ordered NNF tableau. ψ is marked as having had a rule applied to it on B.

Dis Assume T is already an ordered NNF tableau for ϕ, B is a branch of T, $\psi{=}\psi_1 \vee \cdots \vee \psi_n$ is on B, ψ has an ordered link into B and it had not yet a rule applied to it on B. Then create n new branches below B each of which contains a single new node and is labelled with one of the ψ_i. The resulting tableau again is an ordered NNF tableau. ψ is marked as having had a rule applied to it on B.

Univ Assume T is already an ordered NNF tableau for ϕ, B is a branch of T, $(\forall x)\phi(x)$ is on B, t is a ground term, $\phi(t)$ does not occur on B and it has an ordered link into B. Then B is extended by a new node which is labelled with $\phi(t)$. The resulting tableau again is an ordered NNF tableau.

Remark

1. Unlike in CNF tableaux, all formulas ever to be expanded are present on the initial tableau as subformulas of the initial formula.
2. In general, ordered link information is needed for all subformulas of the initial formula (as there are only linearly many subformulas in each formula this is not prohibitive).
3. A formula might have an ordered link to itself. For instance, the first rule application in an ordered tableau with an unsatisfiable initial formula is triggered this way.
4. The generalization from skolemized NNF input to arbitrary skolemized formulas (without $\leftrightarrow, \not\leftrightarrow$) is straightforward: it suffices to use uniform notation and to pay attention to polarity of subformulas in the definitions of c-paths, d-paths, and links. We restricted ourselves to the NNF case to avoid technicalities which only obscure the real problems at issue. The details can safely be left to the reader.

Example 3 We show an ordered NNF tableau for a set of first-order formulas in Figure 1. Each formula (but the first five formulas, which constitute the initial tableau) begins with two numbers i:j, where i is the number of its premise and j the number of the formula. To keep the tableau representation small an application of a *Univ* rule followed by an application of a *Dis* rule is denoted as a single step. We used the following A-ordering:

$P(t_1, \ldots, t_n) <_A P'(t'_1, \ldots, t'_n)$ iff $P < P'$ or $P = P'$ and $t_i <_A t'_i$ for all $1 \leq i \leq n$. P, P' are arbitrary function or predicate symbols with $s < r < q < p$ and function symbols are ordered alphabetically. All other terms are incomparable.

Maximal literal occurrences in non-literals are framed.

There are two ordered links: $(p(c), \neg p(x))$ between the formulas 5 and 2, $(p(c), \neg p(x))$ between 5 and 3. Other formulas can only be expanded if the complement of their maximal literals is on the branch.

This leaves only formulas number 2, 3, and 5 as candidates for the first step. It is a common strategy to consider ground formulas first, thus we take number 5. On the left branch we are left with formula 2 or 3 as a choice, because formulas number 6 does not produce any new ordered links. If we expand formula 3, then the left of the new branches is closed and formula 9 which is new on the right branch again causes no new ordered links to appear, because r does not occur maximally anywhere. If we employ a fair selection strategy for universal formulas we are, therefore, left with formula 2 as the only possibility for expansion. From this point onwards no restriction is achieved by orderings, because now q appears on both remaining open branches which has ordered links to formulas 1 and 4, hence all formulas are eligible for expansion.

Theorem 4 Let ϕ be a closed NNF formula. If ϕ is unsatisfiable, then there is a finite closed ordered NNF tableau for ϕ.

Proof Let T be the (usually infinite) tableau constructed by the following strategy: If the rules *Con* or *Dis* are applicable, apply these first. There are only finitely many of them at each time. If several *Univ* rules but no *Con* or *Dis* rules applicable, take the smallest instance with respect to a given enumeration of ground formulas. Obviously, each open branch of T is an ordered Hintikka set. Since ordered Hintikka sets are satisfiable and the root formula of T is unsatisfiable, such a branch cannot exist, hence all branches of T are closed. König's lemma guarantees that there is a closed finite subtableau of T.

As in the ground clause case, proof confluence follows by the fact, that each ordered tableau can be extended to a "saturated" ordered tableaux whose branches constitute ordered Hintikka sets.

It is crucial for completeness that during the extension of a tableau literals in the newly generated formulas can become maximal. The more restrictive version of the tableau extension rule where maximality of literals is evaluated relative to the *initial formula* is easily seen to be incomplete as it can produce subformulas without maximal literals.

Consider the unsatisfiable formula $(\neg A \vee \neg C) \wedge ((A \wedge B) \vee C)$. If C is maximal, then the subformula $A \wedge B$ does not contain a maximal literal anymore and cannot be expanded, hence there is no closed tableau.

The rôle and definition of ordered Hintikka sets (and the branches corresponding to them) closely parallels that of clause sets saturated with respect to application of certain resolution

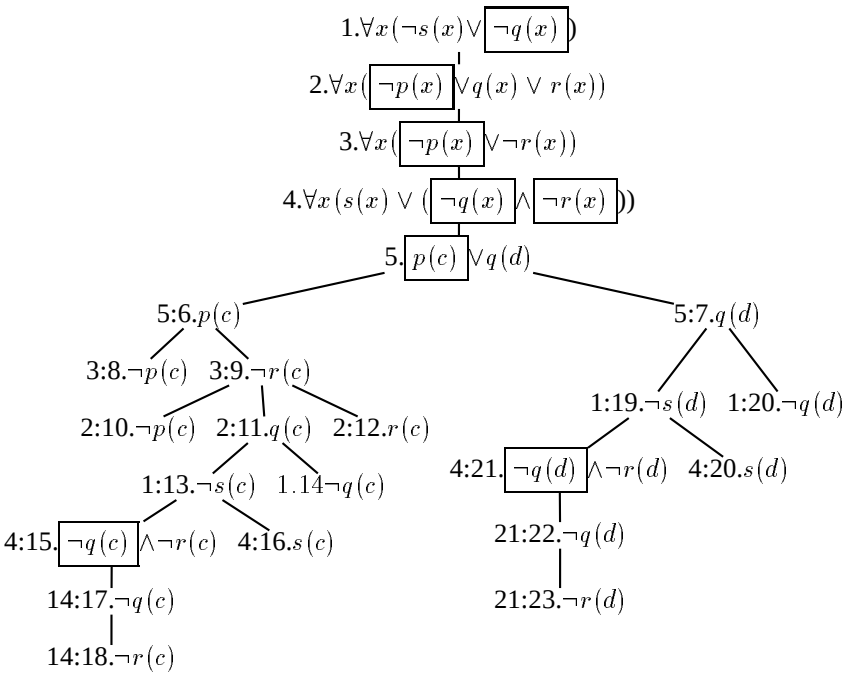


Figure 1: A closed A-ordered tableau for the formulas numbered 1.–5. (Pelletier No. 24).

rules in the framework of Bachmair & Ganzinger [2]. Some differences arise, however, because A-orderings need not be total, whereas in [2] total and well-founded orderings are considered.

5.2 Ordered Free Variable Tableaux

So far we presented essentially Smullyan's version of tableaux [11], where instantiations of universal formulas are guessed. Tableaux implementations work either with an enumeration of ground instances or with free variables and unification. In both cases, the ordering relations among literals of the formula to refute have to be analyzed before starting the proof. A first step determines the maximal literal occurrences in each subformula. Since the maximal literal occurrences of a formula are also maximal literal occurrences of subformulas of the formula, maximal occurrences can be detected recursively. In a second step the link table is computed. All formulas occurring during a proof are instantiations of subformulas of the formula to refute. According to lemma 1, the maximal literal occurrences of the instantiations correspond to maximal literal occurrences of the original formula. Thus, no new links have to be generated during the proof. Some links might, however, become obsolete in the instantiations, either because the linked literals cannot be unified any longer or one of these literal occurrences ceased to be maximal. In a ground instance enumeration procedure these two points can easily be checked before each expansion step.

More interesting are, however, free variable first-order tableaux [6]. We discuss a ver-

sion which uses fair formula selection and backtracking over applied substitutions (as implemented, without A-orderings, for example, in [3, 4]), because it comprises fairness and backtracking aspects as well. Free variables constitute an implementational problem, because the ordering restriction on a rule application can not be checked at the time the rule is used. Later substitutions may affect the maximal occurrence or complementarity of the linked literal occurrences. As a first approach, one could check before each substitution, whether the tableau is still ordered after the substitution. It is not quite clear yet how this can be implemented efficiently. An approximation might be achieved by using syntactic term constraints like the constraints used for efficient implementation of regularity [9].

A different approach focuses more on the *links* used to expand a formula. It is easier to enforce the ordering restriction with respect to the used links than to check at each stage, whether a tableau is still ordered or not. In other words, whenever a link is used, one has to guarantee that future substitutions do not violate the ordering restrictions of the used link. This is easy to do as far as complementarity is concerned. If one applies the link unifier to the tableau, whenever a formula is expanded, complementarity cannot be destroyed later. This approach has three immediate consequences:

- Instead of using new variables to instantiate universally quantified formulas, terms obtained from the link unifier are used. They might contain new free variables obtained from renamed bound variables.
- Instead of fair **formula** selection, **links** must be selected in a fair way.
- Backtracking occurs for all substitutions, in particular for those performed during expansion steps.

This still leaves open the problem that through certain substitutions literals may cease to be maximal; the latter can be guaranteed only by constraints:

Consider the A-ordering used in Example 3. In this A-ordering the literals $p(x, f(x))$ and $p(a, y)$ are incomparable.

Assume $p(x, f(x))$ and $p(a, y)$ occur on the same d-path and $p(x, f(x)$ was used in a link. Each following substitution must preserve the maximality of $p(x, f(x)$. Therefore, one must generate a constraint of the form ($x\sigma \not<_A a$ or $f(x\sigma) \not<_A y\sigma$) which has to be checked whenever a substituion σ is applied to x or y.

Obviously, constraint generation depends on the A-ordering. Many problems do not require a lot of constraints. As long as with respect to the chosen A-ordering every d-path has exactly one maximal literal, maximal literal occurrences stay maximal after arbitrary substitutions. A suitable A-ordering can achieve this effect for many problems. This can actually be used as a guideline for choosing A-orderings.

In the light of the previous discussion, it seems practicable to avoid constraints as far as possible. It is easy to implement a procedure without constraints, which applies substitutions only for branch closure and the *Univ* rule. Figure 1 shows, that even this weak version of ordered tableaux might be interesting: obviously, no free variables can be introduced, whenever formula 5 is expanded first. For this particular example the proposed procedure neither has the disadvantages of backtracking nor those of a ground instance enumeration tableaux.

The implementation and evaluation of various versions of free variable ordered tableaux will be the topic of a forthcoming paper.

Acknowledgements

We would like to thank Leo Bachmair who pointed out to us that a suitable adaption of the techniques in [2] can be used to establish the completeness of ordered ground clause tableaux.

References

[1] P. B. Andrews. Theorem proving through general matings. *JACM*, 28:193–214, 1981.

[2] L. Bachmair and H. Ganzinger. Rewrite-based equational theorem proving with selection and simplification. *Journal of Logic and Computation*, 4(3):217–247, 1994.

[3] B. Beckert, S. Gerberding, R. Hähnle, and W. Kernig. The many-valued tableau-based theorem prover ${}_3T^AP$. In D. Kapur, editor, *Proc. 11th Conference on Automated Deduction, Albany/NY*, pages 758–760. Springer LNAI 607, 1992.

[4] B. Beckert and J. Posegga. leanT^AP: Lean tableau-based theorem proving. extended abstract. In A. Bundy, editor, *Proceedings, 12th International Conference on Automated Deduction (CADE), Nancy, France*, volume 814 of *LNCS*, pages 793–797. Springer-Verlag, 1994.

[5] W. Bibel. *Automated Theorem Proving*. Vieweg, Braunschweig, second revised edition, 1987.

[6] M. C. Fitting. *First-Order Logic and Automated Theorem Proving*. Springer, New York, 1990.

[7] S. Klingenbeck and R. Hähnle. Semantic tableaux with ordering restrictions. In A. Bundy, editor, *Proc. 12th Conference on Automated Deduction CADE, Nancy/France*, LNAI 814, pages 708–722. Springer Verlag, 1994.

[8] R. Letz. *First-Order Calculi and Proof Procedures for Automated Deduction*. PhD thesis, TH Darmstadt, June 1993.

[9] R. Letz, J. Schumann, S. Bayerl, and W. Bibel. SETHEO: A high-perfomance theorem prover. *Journal of Automated Reasoning*, 8(2):183–212, 1992.

[10] N. V. Murray and E. Rosenthal. Dissolution: Making paths vanish. *Journal of the ACM*, 3(40):504–535, 1993.

[11] R. M. Smullyan. *First-Order Logic*. Dover Publications, New York, second corrected edition, 1995. First published 1968 by Springer-Verlag.