Restart Tableaux with Selection Function

Christian Pape and Reiner Hähnle

Universität Karlsruhe
Institut für Logik, Komplexität und Deduktionssysteme
76128 Karlsruhe, Germany
{pape,reiner}@ira.uka.de

Abstract. Recently, several different sound and complete tableau calculi were introduced, all sharing the idea to use a selection function and so-called restart clauses: A-ordered tableaux, tableaux with selection function, and strict restart model elimination. We present two new sound and complete abstract tableau calculi which generalize these on the ground level. This makes differences and similarities between the calculi clearer and, in addition, gives insight into how properties of the calculi can be transferred among them. In particular, a precise borderline separating proof confluent from non-proof confluent variants is exhibited.

1 Introduction

In this paper we introduce two new ground[1] tableau calculi, called *restart tableaux* and *strict restart tableaux*. Restart tableaux generalize the recently developed A-ordered tableaux [2] and tableaux with selection function [3], whereas strict restart tableaux subsume strict restart model elimination [1].

All of these calculi can be uniformly described by restricting the usual extension rule of clause tableaux:

1. a selection function (i.e., a mapping from clauses to subsets of their literals) restricts possible connections of clauses used for extension steps;
2. extension steps are either weakly connected (i.e. to any branch literal) or strongly connected (i.e. to a leaf);
3. so-called restart steps (extension steps with certain unconnected clauses) are used to continue proof search, whenever it is not possible to employ connected extension steps.

These abstract features are used by the currently known calculi in differing ways. Our notions (see Section 3.1 below) generalize them all.

Further ingredients of tableau calculi specify to which extent branches are regular (i.e. are free of repetitions) and which literals are permitted for closing branches. It turns out that these latter conditions are determined by those on

[1] A thorough treatment of first-order logic would have doubled the size of the paper, although there are no principal obstacles. See the last section for a few hints to lifting.

restart steps and on the amount of strong connections. In fact there is a direct trade-off between restrictive restarts and preference of strong connections (i.e. high connectivity) and regularity/reduction steps (at least if complete calculi are desired).

In this paper we define abstract tableau calculi whose properties can be adjusted within a wide range while the completeness proof is fully generic. This leaves open the possibility to fine-tune a calculus to each theorem one wants to prove with it.

An important difference between A-ordered tableaux and tableaux with selection function on the one side and restart model elimination on the other is the lack of proof confluency[2] in the latter. We investigate the reasons for this and show that already a very slight liberalization of restart model elimination gives proof confluency. Thus we exhibit a precise proof theoretical borderline separating calculi that are proof confluent from those that are not. Moreover, restart tableaux turn out to be a proof confluent procedure which is extremely close to restart model elimination.

Section 2 states basic notions and Section 3 briefly reviews existing calculi. In Section 4 we define restart tableaux and prove their completeness. A-ordered tableaux and tableaux with selection function are obtained as instances. In Section 5 restart tableaux are modified to strict restart tableaux which are also proven to be complete. Strict restart model elimination is obtained as an instance. At the end we make some brief remarks about lifting to first-order logic.

2 Notation

From a signature Σ (predicate, constant and function symbols) **atoms** and **literals** are constructed using the negation sign $\neg$ as usual. The set of all literals over Σ is denoted by $\mathcal{L}_\Sigma$. We omit the index Σ if no confusion can arise. In this paper we only deal with *ground clauses* that is all atoms are variable free.

A **clause** is a sequence $L_1 \vee \ldots \vee L_n, n \geq 1$ of disjunctively connected literals. $\mathcal{C}$ is the set of all clauses. We write $L \in C$ for short if a literal L occurs in a clause C. $\overline{L}$ is the **complement** of a literal L, i.e. $\overline{A} = \neg A$ and $\overline{\neg A} = A$ if A is an atom.

A **tableau** $\mathfrak{T}$ is an ordered tree where the root node is labeled with *true* or a literal and all other nodes are labeled with literals. For a node n of $\mathfrak{T}$ the clause clauseof(n) is constructed from the literals of the children of n in the order from left to right. A path from the root node to a leaf literal of $\mathfrak{T}$ is called a **branch** of $\mathfrak{T}$. A tableau is **closed** if every branch contains (at least) two complementary literals. We sometimes describe a branch as a finite set of literals. We also often identify branches with the set of literals on them.

[2] A calculus is *proof confluent* if each partial proof for a provable theorem can be completed. Typical well-known examples of non-proof confluent calculi are model elimination and linear resolution.

A branch B is said to be **regular** if (i) every literal of a node of B occurs only once on B and (ii) clauseof(n) is no tautology for every node n of B. A tableau $\mathfrak{T}$ is **regular** if all branches of $\mathfrak{T}$ are regular.

Partial interpretations are associated with consistent sets of ground literals. An interpretation I **satisfies** a ground clause C iff there is an $L \in C$ with $L \in I$. I is a **model** for a set S of clauses iff I satisfies all clauses of S.

3 Tableaux with Restarts and Selection Function

In this section we rehash definitions of tableaux with selection function [3], of A-ordered tableaux [2], and of restart model elimination [1]. For motivation and more examples, see the papers cited. Completeness of these calculi is obtained later from more general results and not re-stated here. We only give ground versions. Note that in the following the various notions of restart and of selection function slightly differ among the calculi. We unify them later.

3.1 Tableaux with Selection Function

Definition 1. A **selection function** is a total function f from ground clauses to sets of literals such that $\emptyset \neq f(C) \subseteq C$ for all $C \in \mathcal{C}$.

f is used to restrict connections between clauses to literals selected by f. Unrestricted extension steps, so called restarts, are only allowed with clauses that have at least a connection to another clause.

Definition 2. Let $S \subset \mathcal{C}$, f a selection function, B a tableau branch. C is a **restart clause (of S)** iff there is a $D \in S$ and a literal L such that $L \in f(D)$ and $\overline{L} \in f(C)$. C has a **weak connection via f and L into B** iff there is an $L \in B$ such that $\overline{L} \in f(C)$.

Definition 3. Let $S \subset \mathcal{C}$, f a selection function, then a **tableau with selection function f for S** is a regular clause tableau $\mathfrak{T}$ such that for every node n of $\mathfrak{T}$ the clause clauseof(n) (i) has a weak connection via f into the branch ending in n or (ii) it is a restart clause.

Example 1. Fig. 1 shows a closed tableau with selection function f for the set of ground clauses $S = \{\underline{\neg A} \vee \neg B, \underline{A} \vee B, \underline{A} \vee \neg C, \underline{B} \vee C, \underline{B} \vee \neg C\}$ (f selects the underlined literals). The first three clauses of S can be used for restarts. The solid lines correspond to extension steps (weak connection) and the dashed lines to other closures.

3.2 A-ordered tableaux

Definition 4. An **A-ordering** is an irreflexive, transitive relation $\prec$ on atoms, such that for all atoms A, B: $A \prec B$ implies $A\sigma \prec B\sigma$ for all substitutions σ (**stability wrt substitutions**).

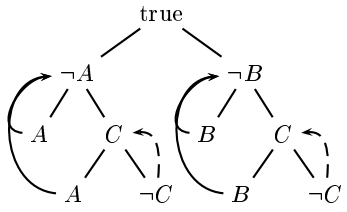

Fig. 1. Tableau with selection function

A-orderings can be extended to literal orderings via $L \prec L'$ iff $\mathrm{atom}(L) \prec \mathrm{atom}(L')$ (where $\mathrm{atom}(A) = \mathrm{atom}(\neg A) = A$). As in ordered resolution connections between clauses are restricted to literals that occur $\prec$-maximally in these clauses.

Definition 5. A literal L_j occurs **$\prec$-maximally** in a clause $L_1 \vee \ldots \vee L_n$ iff $L_j \not\prec L_i$ for all $i = 1, \ldots, n$.

A clause $C = L_1 \vee \ldots \vee L_n$ has a **$\prec$-maximal connection** to a clause $C' = L'_1 \vee \ldots \vee L'_{n'}$ iff $L_i = \overline{L'_j}$ for some $1 \leq i \leq n, 1 \leq j \leq n'$, L_i occurs maximally in C, and L'_j occurs maximally in C'. If both $C, C' \in S$ then C is a **restart clause** of S.

A clause C has a **maximal connection into a set of literals** B iff C has a maximal connection to a clause consisting of a single literal of B.

A-ordered tableaux are regular clause tableaux with the restriction that extension steps are only possible with clauses that have a maximal connection into the branch they extend or to another clause of S:

Definition 6. Let $\prec$ be an A-ordering and S a set of ground clauses. A **$\prec$-ordered clause tableau for** S is a regular clause tableau $\mathfrak{T}$ for S such that for every node n of $\mathfrak{T}$ the clause clauseof(n) (i) has a $\prec$-maximal connection into the branch ending in n or (ii) it is a restart clause.

Every A-ordering $\prec$ induces a selection function $f_\prec$ on literals (such that $f_\prec(C)$ are exactly the $\prec$-maximal literals of C), hence A-ordered tableaux can be seen as an instance of tableaux with selection function, see [3].

Example 2. Let S be as in Ex. 1. For the A-ordering $A > B > C$ obviously $f_< = f$ on S, where f is as in Ex. 1. Thus, Fig. 1 constitutes as well an A-ordered tableaux for S and $<$.

3.3 (Strict) Restart Model Elimination

In contrast to tableaux with selection function in restart model elimination (RME) the selection function f (i) applies only to non-negative clauses and

(ii) selects exactly one positive literal. As a consequence, a clause is never connected via f to another clause and, thus, there are no restart clauses in the sense of tableaux with selection function and A-ordered tableaux. Instead, the role of restart clauses is taken on by negative clauses.

Definition 7. A **selection function** is a total function f from non-negative ground clauses to literals, such that $f(C)$ occurs in C and is positive. Every negative clause is a **restart clause**.

RME is a refinement of model elimination [5], hence all connections are restricted to the leaf literals of a branch.

Definition 8. Let f be a selection function, B a tableau branch, $C \in \mathcal{C}$. If L is the leaf literal of B and $\overline{L} = f(C)$, then C has a **strong connection via f and L into B**.

A clause that has a connection (either weak or strong) into a branch B, where it is used gives rise to immediate closure of at least one of the new branches, namely the one with a selected (or maximal) literal L whose complement occurs on B. In this case we say that L is a **connection literal**. Closed branches not closed by a connection literal are said to contain a **reduction step**.

Most implementations of theorem provers based on model elimination and on RME are using Prolog Technology Theorem Proving (PTTP) [6]. In PTTP a set of clauses is compiled into a Prolog program such that every literal of a clause is the head of a Prolog clause (so called contrapositives). In RME only one contrapositive of each non-negative clause needs to be used in a PTTP implementation. This can lead to a significant reduction of the search space during the proof.

Unfortunately, the above restriction in combination with regularity [4] leads to an incomplete calculus (see Ex. 3). To restore completeness, the regularity condition of RME has to be relaxed:

Definition 9. Given a tableau $\mathfrak{T}$ and a subbranch[3] $\langle L_{n_0}, \ldots, L_{n_i} \rangle$ of $\mathfrak{T}$. Then $\langle L_{n_1}, \ldots, L_{n_i} \rangle$ is called a **block (corresponding to a restart clause C)** iff clauseof$(n_0) = C$ is a restart clause, for no $j = 1, \ldots, i-1$ is clauseof(n_j) a restart clause, and

1. either n_i is a leaf of $\mathfrak{T}$ or
2. clauseof(n_i) is a restart clause.

A branch B is **blockwise regular** iff every block of B is regular.

Definition 10. A **restart model elimination (RME) tableau** for a set S of ground clauses is a clause tableau $\mathfrak{T}$ for S such that:

1. For every node n of $\mathfrak{T}$ (i) clauseof(n) has a strong connection into the branch ending in n, or (ii) clauseof(n) is a restart clause and the label of n is a positive literal.

[3] The sequence of labels of a contiguous subset of the nodes in a branch.

2. Every branch is (i) regular wrt positive literals (**positive regular**) and (ii) blockwise regular.

Example 3. Let S be as in Ex. 1. As f selects exactly one positive literal in each non-negative clause f is a suitable selection function for a restart model elimination tableau, if we ignore it on the only negative clause. This clause is by Def. 7 also the only restart clause. Fig. 2 shows an RME tableau for S. Due to symmetry in S the open branch on the right can be closed in a similar way as the left one. Note that for completeness of the calculus it is essential to permit multiple occurrences of negative literals on a tableau branch (condition 2.(i) in Def. 10).

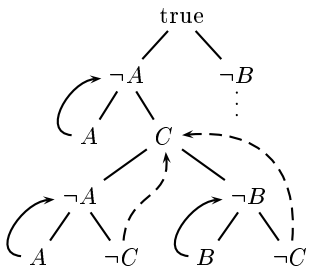

Fig. 2. A restart model elimination tableau

In a PTTP implementation reduction steps usually slow down the speed of the theorem prover, because they involve search among the literals on the current branch. The following modification of RME restricts reduction steps to negative leaf literals which gains some benefit in a PTTP implementation.

Definition 11. A branch B in a **strict restart model elimination tableau** for a set S of ground clauses is **closed** if its leaf literal L is (i) a connection literal or (ii) is negative and $\overline{L} \in B$.

4 Restart Tableaux with Selection Function

In RME the selection function selects exactly one positive literal, while in tableaux with selection function all literals can be selected. In restart tableaux we use a notion of selection function which generalizes Defs. 1, 7. This version of selection function is meant in the remainder of the paper if not explicitly said otherwise.

Definition 12. A **selection function** is a total function f from ground clauses to sets of literals such that $f(C) \subseteq C$ for all $C \in \mathcal{C}$.

Next we unify the various notions of restart clause.

Definition 13. Let $S \subset \mathcal{C}$, f a selection function, B a tableau branch. C is a **restart clause (of S)** iff (i) there is a $D \in S$ and a literal L with $L \in f(D)$ and $\overline{L} \in f(C)$ or (ii) $f(C) = \emptyset$.

In tableaux with selection function a non-restart clause C can extend a tableau branch B only if a selected literal of C is complementary to a literal on B, whereas in restart model elimination C has to be complementary to the leaf literal on B. Therefore, a unifying calculus has to deal with both kinds of connections. We merely repeat (parts of) Defs. 2, 8:

Definition 14. Given a clause C, a tableau branch B, and a selection function f. Then C has a **weak connection via f and L into** B iff there exists $L \in B$ with $\overline{L} \in f(C)$. It has a **strong connection via f and L into** B iff $\overline{L} \in f(C)$ for the leaf literal L of B. In both cases we say that C is **connected to B via f (and L)**.

For a given set S of clauses we fix in advance whether a clause of S can extend a tableau branch with a weak or with a strong connection. This leads to a partition of S into two disjoint subsets S_w and S_s, such that

- the clauses of S_w can extend a tableau branch B if they have a weak connection into B,[4] and
- the clauses of S_s can extend B only if they have a strong connection into B.

It will become necessary to compare blocks (cf. Def. 9) wrt to their literal sets. In the following definition and elsewhere we handle blocks (which are defined as sequences of literals) as sets of literals without the danger of confusion.

Definition 15. Two blocks b and b' are **distinct** iff neither $b \subseteq b'$ nor $b' \subseteq b$.

In the next definition a somewhat complicated notion of regularity is used which is motivated as follows: A clause $C \in S_w$ can be used to extend a branch B whenever a selected literal of C is complementary to a literal on B. The complement of the connection literal can be anywhere in the current block. Now, in the case of a strong connection two different clause $C, D \in S_s$ may be required to extend a branch via the same selected literal (in different blocks) which, therefore, occurs twice on this branch, cf. Figure 2. On the other hand, if B contains a literal L such that its complement $\overline{L}$ is never selected in S_s, then a clause of S_s can never extend a branch with leaf literal L and branch-wise regularity can be enforced wrt such literals. Formally, these literals are defined as $L_{S,f} = \mathcal{L} - \bigcup_{C \in S_s} \{\overline{L} | \, L \in f(C)\}$.

If we choose, for example, $S_s = \emptyset$ then $L_S = \mathcal{L}$ which implies the usual regularity condition as of tableaux with selection function. If $S_w = \emptyset$ and the selection function only selects positive literals, then L_S contains at least all

[4] Note that every strong connection is also a weak connection.

positive literals. This is the regularity condition of restart model elimination, see Def. 10 2.(i).

The above considerations are summarized and formally expressed in the following definition.

Definition 16. $S_w \cup S_s$ is any partition of $S \subset \mathcal{C}$ and f a selection function. A **restart tableau (RT) for S and f** is a clause tableau $\mathfrak{T}$ for S such that:

1. For every node n of $\mathfrak{T}$ one of the following holds:
   (a) clauseof(n) $\in S_s$ has a strong connection via f into the block ending in n *(strong extension step)*;
   (b) clauseof(n) $\in S_w$ has a weak connection via f into the block ending in n *(weak extension step)*;
   (c) clauseof(n) is a restart clause and it is not possible to extend the block above n with a strong or a weak extension step.
2. For every branch B of $\mathfrak{T}$ all of the following hold:
   (a) B is blockwise regular;
   (b) B is regular wrt $L_{S,f}$;
   (c) B contains only distinct blocks.

The definition of closure of RT tableaux is as usual (i.e. not as in Def. 11).

Definition 17. A RT $\mathfrak{T}$ for a ground clause set S is called **saturated** iff there exists no RT $\mathfrak{T}'$ for S such that $\mathfrak{T}$ is a proper subtree of $\mathfrak{T}'$.

Lemma 18. *Let S be a finite set of ground clauses. Then every branch of a saturated RT $\mathfrak{T}$ for S is finite.*

Proof. First note that each block of a branch B of $\mathfrak{T}$ is finite by Def. 16.2a. By Def. 16.2c there is a finite number of possible different blocks in B which proves the claim. □

Theorem 19. *Given a finite unsatisfiable set S of ground clauses and a selection function f. Then there exists a closed RT for S and f.*

Proof. By regularity we can assume that S contains no tautologies.

Assume there is no closed RT for S. Then, by Lemma 18, for every saturated RT $\mathfrak{T}$ for S an open and finite branch B exists. The literals of B constitute a partial interpretation I of S (via $I = B$).

Let S' be the set of all clauses from S not satisfied by I. As $\mathfrak{T}$ is saturated there is no restart clause in S', otherwise such a clause could be used to extend B by Def. 16.1c.

J is the partial interpretation that satisfies the selected literals of clauses in S'. J is well-defined, otherwise two clauses of S' would be connected via f and hence restart clauses.

$I \cup J$ is well-defined, too: If not, there are literals $L \in I$ and $\overline{L} \in J$. We distinguish two cases: either (i) $C \in S_w$ or (ii) $C \in S_s$, where $C = \overline{L} \vee L_1 \vee \ldots \vee L_n$ and $\overline{L} \in f(C)$. We show that in both cases $\mathfrak{T}$ is not saturated.

case (i) here we have $C \in S_w \cap S'$ and C has a weak connection into a block b of B. Any weak extension step (Def. 16.1b) with C produces several new branches with leaf literals not already on B (otherwise C would not be in S'). Therefore, regularity is maintained.

case (ii) We show how to initiate a restart on the leaf of B and to extend B with a new block, such that regularity still holds.
First select a block $\langle L'_{n_1}, \ldots, L'_{n_m} \rangle$ of B with nodes $n_1, \ldots, n_m$ and beginning with a restart clause D such that L occurs in this block, say, $L = L'_{n_i}$, $1 \leq i \leq m$. Then extend B with D and clauseof(n_1) up to clauseof(n_{i-1}) and call the resulting block b. This is possible, simply because it was possible earlier in B, but we must take care to make b distinct from all other blocks. This is done by extending b with C (recall that in the present case $C \in S_s \cap S'$). This strong extension step (Def. 16.1b) generates n new distinct blocks $b_1, \ldots, b_n$ (see right part of Fig. 3) each of which satisfies Def. 16.2a–16.2c:

(a) Every block b_i is regular up to L_i because it is a prefix of a regular block and B does not contain L_i, otherwise I satisfies D.

(b) The b_i do not contain a literal $L' \in L_{S,f}$: Such a literal causes a restart and L' would have to be the last node of b which then cannot be extended by C. Thus, the new branches are regular wrt L_S.

(c) each b_i is a new distinct block on B, because it contains a literal L_i not occurring on B (otherwise C would not be in S').

We conclude that $I \cup J$ is a model of S contradicting its unsatisfiability. □

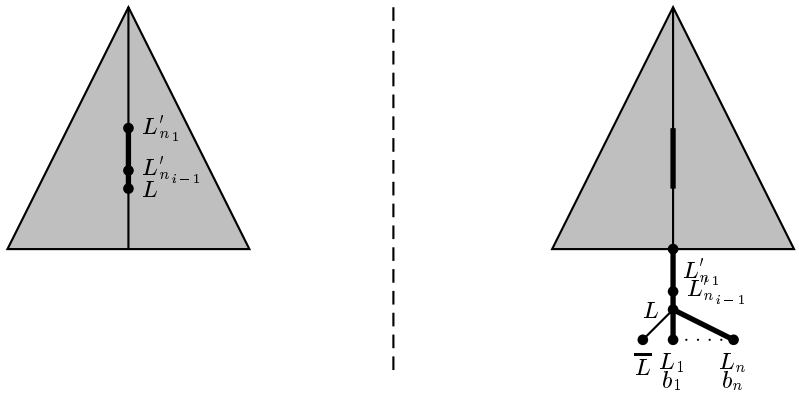

Fig. 3. Initiate a new restart (see text)

Note that, as a consequence of our proof, RT are proof confluent.

Ground completeness of tableaux with selection function follows easily from Theorem 19:

Corollary 20 [3]. *For each finite unsatisfiable set S of ground clauses and selection function f (in the sense of Def. 1) there exists a closed tableau with selection function f for S.*

Proof. Set $S_w = S$. Then L_S contains all literals from S and, hence, every branch of a restart tableau has to be regular wrt to all literals, which holds also for tableaux with selection function. □

Restart tableaux are impossible to instantiate to restart model elimination, because restart model elimination is known not to be proof confluent.[5] On the other hand, if we allow restarts on negative leaf literals as well (and call the resulting calculus **unrestricted restart model elimination**), then we obtain:

Corollary 21. *For each finite unsatisfiable set S of ground clauses and selection function f (in the sense of Def. 7) there exists an unrestricted restart model elimination tableau for S and f.*

Proof. In RME all non-restart clauses must be part of strong extension steps, therefore set $S_s = S$. f selects only positive literals from a clause or none if the clause is negative, hence $L_{S,f}$ contains at least all positive literals. For restart tableaux these settings imply that each branch is blockwise regular and regular wrt to positive literals. □

It is remarkable that merely admitting positive literals in restart steps decides whether restart model elimination is proof confluent or not.

5 Strict Restart Tableaux

Restart tableaux, although very close in spirit to restart model elimination, bear a small, but crucial, difference to the latter: restarts are restricted to positive leaf literals. In addition, in strict restart model elimination reduction steps with negative leaf literals are excluded.

In this section we modify restart tableaux to a calculus of which (strict) restart model elimination is an instance.

Recall the definition of the literal set $L_{S,f}$ in Section 4 which controls regularity. Let $L_R \subset L_{S,f}$ be any set that does not contain complementary literals.

Definition 22. $S_w \cup S_s$ is any partition of $S \subset \mathcal{C}$ and f a selection function. A **strict restart tableau (SRT) for S and f** is a clause tableau $\mathfrak{T}$ for S such that:

1. For every node n of $\mathfrak{T}$ one of the following holds:
   (a) clauseof(n) $\in S_s$ has a strong connection via f into the block ending in n *(strong extension step)*;

[5] Consider the unsatisfiable set $\{\neg A, A \vee \neg B, A \vee \neg C, C\}$ and the partial tableau generated by the first two clauses. The open branch with negative leaf $\neg B$ cannot be closed or extended.

(b) clauseof(n) $\in S_w$ has a weak connection via f into the *branch* ending in n *(weak extension step)*;
(c) clauseof(n) is a restart clause and the literal of n is not in $\{\overline{L} | \, L \in L_R\}$.

2. For every branch B of $\mathfrak{T}$ both of the following hold:
   (a) B is blockwise regular;
   (b) B is regular wrt L_R.

Definition 23. A branch B of an SRT is **closed** iff its leaf literal L is (i) a connection literal or (ii) $L \notin L_R$ and there is $\overline{L} \in B$. The latter is called a *strict reduction step*.

In comparison to restart tableaux one notes two important relaxations: weak connections need not be local to a block anymore and blocks may be identical. Moreover, restarts must be permitted even when extensions steps are still possible. For this reason RT and SRT are not instances of each other.

Some instances of strict restart tableaux are proof confluent, others are not, so there is no way to obtain a completeness proof based on saturation as for restart tableaux. One way to view strict restart tableau proofs is as a kind of normal form for restart tableau proofs. The proof below is by a tableau transformation that computes exactly this normal form, thus establishing completeness of strict restart tableaux. Note that this does not impose any assumption on proof confluency of strict restart tableaux as we start out with a closed tableau.

Unfortunately, the transformation destroys regularity conditions Def. 16.2b and 16.2c, because it copies parts of the proof tree. The following lemma shows that at least Def. 22.2b can be regained. Details of proofs had to be left out due to space restrictions, but the proofs are rather tedious than deep, anyway.

Lemma 24. *For each closed strict restart tableau $\mathfrak{T}$ for S which is blockwise regular but not necessarily regular wrt L_R there exists a closed strict restart tableau $\mathfrak{T}'$ for S which is also regular wrt L_R.*

Proof. The proof is by a careful analysis showing that critical occurrences of duplicate literals can be deleted without changing the rest. □

Theorem 25. *For any finite unsatisfiable set $S \subset \mathcal{C}$ and selection function f there exists a closed strict restart tableau for S and f.*

Proof. Two simple inductions (i) eliminate critical restarts by exchanging blocks, (ii) remove critical reduction steps by copying suitable subtrees. Then Lemma 24 is applied. The details are straightforward, though tedious and technical. □

Corollary 26 [1]. *For any finite unsatisfiable set $S \subset \mathcal{C}$ and selection function f (in the sense of Def. 7) there exists a strict restart model elimination tableau for S and f.*

Proof. Set $S_w = \emptyset$ and $S_s = S$. As f selects only positive literals in clauses of S, $L_{S,f}$ consists of at least all positive literals, so take as $L_R \subset L_{S,f}$ the set of all

positive literals. With these settings a strict restart tableau is blockwise regular, positive regular, and reduction steps are only allowed on negative literals; thus it is a strict restart model elimination tableau. □

As noted in [1] completeness of RME can be derived from this result as well.

6 Outlook

We introduced two new abstract, sound and complete tableau calculi that generalize other calculi using restart clauses: A-ordered tableaux, tableaux with selection function, and restart model elimination. This gives a whole spectrum of new proof procedures that can be fine-tuned at the selection function they use and at the desired amount of connectivity in proofs. We show explicitly how regularity and restrictions on reduction steps are influenced by the choice of these parameters. Such knowledge can in the future provide the basis for computing an instance of a proof procedure optimized for solving a given problem.

Our framework also helps to determine proof confluency of restart calculi. In particular, restart tableaux which are proof confluent can be instantiated such that they are extremely close to restart model elimination which is not.

Lifting of (strict) restart tableaux to first-order logic can be done as usual if the selection function, resp., the A-ordering is stable wrt substitutions, cf. Def. 4. In addition, a similar property has to be stated for the partition of S into weakly and strongly connected clauses. In an optimal implementation this approach leads to constraints for the regularity condition of restart tableaux. Like in [3] for tableaux with selection function one can make a compromise and get rid of the constraints in a slightly less restrictive proof procedure which admits a much weaker assumption than stability wrt substitutions: stability wrt variable renaming.

References

1. P. Baumgartner and U. Furbach. Model Elimination without Contrapositives and its Application to PTTP. *Journal of Automated Reasoning*, 13:339–359, 1994.
2. R. Hähnle and S. Klingenbeck. A-ordered tableaux. *Journal of Logic and Computation*, 6(6):819–834, 1996.
3. R. Hähnle and C. Pape. Ordered tableaux: Extensions and applications. In D. Galmiche, ed., *Proc. Int. Conf. on Analytic Tableaux, Nancy*, LNCS. Springer-Verlag, 1997, `ftp://129.13.31.2/pub/haehnle/HaehnlePape.ps.gz`.
4. R. Letz, J. Schumann, S. Bayerl, and W. Bibel. SETHEO: A high-perfomance theorem prover. *Journal of Automated Reasoning*, 8(2):183–212, 1992.
5. D. W. Loveland. A linear format for resolution. In *Proc. IRIA Symp. Automatic Demonstration*, pages 147–162, Versailles, France, 1968. Springer-Verlag. Reprinted in: J. Siekmann and G. Wrightson, editors. *Automation of Reasoning: Classical Papers in Computational Logic 1967–1970*, volume 2. Springer-Verlag, 1983.
6. M. E. Stickel. A prolog technology theorem prover. In E. Lusk and R. Overbeek, editors, *Proc. 9^{th} International Conference on Automated Deduction*, pages 752–753. Springer LNCS, New York, 1988.

This article was processed using the LaTeX macro package with LLNCS style