Linear and Unit-Resulting Refutations for Horn Theories [*]

Peter Baumgartner
University of Koblenz
Institute for Computer Science
Rheinau 1
56075 Koblenz
Germany

E-mail: peter@informatik.uni-koblenz.de

August 31, 1995

Abstract. We present a new transformation method by which a given Horn theory is transformed in such a way that resolution derivations can be carried out which are both linear (in the sense of Prologs SLD-resolution) *and* unit-resulting (i.e. the resolvents are unit clauses). This is not trivial since although both strategies alone are complete, their naïve combination is not. Completeness is recovered by our method through a completion procedure in the spirit of Knuth-Bendix completion, however with different ordering criteria. A powerful redundancy criterion helps to find a finite system quite often.
The transformed theory can be used in combination with linear calculi such as e.g. (theory) model elimination to yield sound, complete and efficient calculi for full first order clause logic over the given Horn theory.
As an example application, our method discovers a generalization of the well-known linear paramodulation calculus for the combined theory of equality and strict orderings.
The method has been implemented and has been tested in conjunction with a model elimination theorem prover.

1. Introduction

In proving mathematical theorems a problem can often be divided into two parts: the one part, called the *theory*, describes common knowledge about some problem domain (e.g. equality, orderings, set theory, arithmetic). The other part consists of the *hypothesis* and the concrete *theorem* to be proved. This distinction can be modeled within automated theorem proving by *theory reasoning calculi* such as theory resolution (Stickel, 1985) or theory model elimination (Baumgartner, 1992a; Baumgartner, 1994). There, a "foreground reasoner" (such as model elimination) is coupled with a "background reasoner" for reasoning within the theory. In this scenario, it would be most useful to transform a theory "once and for all" into a background reasoner which can process the theory far more efficiently than it would be possible by supplying the theory axioms as input clauses. Consider for example a transitivity axiom such as $\neg(x < y) \vee \neg(y < z) \vee (x < z)$ for strict orderings. This axiom can be used e.g. in ordinary resolution or model elimination in almost any proof state and leads to an explosion of the search space.

It is the purpose of this paper to define a new and general technique, which transforms a given Horn theory into an inference system for background reasoning within theory

[*] This research was sponsored by the "Deutsche Forschungsgemeinschaft (DFG)" within the "Schwerpunktprogramm *Deduktion*".

reasoning calculi. Currently, the transformation technique is tailored towards the use with *linear, goal-sensitive* (Plaisted *et al.*, 1993) calculi such as model elimination (Loveland, 1968) or linear resolution (Loveland, 1970). Our interest in linear calculi comes from their successful application in automated theorem proving (see e.g. (Baumgartner and Furbach, 1994b; Stickel, 1989; Stickel, 1990; Letz *et al.*, 1992; Astrachan and Stickel, 1992) for descriptions of running systems; the calculus of model elimination was introduced by Loveland (Loveland, 1968; Loveland, 1978), more recent variants are described in (Plaisted, 1990; Bollinger, 1991; Stickel, 1991; Baumgartner, 1992a; Baumgartner and Furbach, 1994a; Letz *et al.*, 1993)). In this paper we will concentrate on the transformation technique alone; its place within e.g. theory model elimination is briefly described in Section 1.1 below.

Our method works by saturating a Horn clause set $\mathcal{T}$ under several deduction operations until only redundant consequences can be added. The resulting (possibly infinite) system $\mathcal{I}_\infty(\mathcal{T})$ enjoys the following completeness property: for every (minimal) $\mathcal{T}$-unsatisfiable literal set $\mathcal{L}$ and every literal $G \in \mathcal{L}$ there exists a *linear* resolution refutation of $\mathcal{I}_\infty(\mathcal{T}) \cup \mathcal{L}$ with goal literal G, i.e. G is processed stepwise until the empty clause is derived, *and*, all inferences are done in a *unit-resulting* way, i.e. in an n-literal parent clause at least $n-1$ literals have to be simultaneously resolved against $n-1$ complementary unit clauses in order to carry out an inference.

Thus our technique — called *linearizing completion* — is a device for combining the unit-resulting strategy of resolution (McCharen *et al.*, 1976) with a linear strategy à la Prolog in a refutationally complete way (See e.g. (Stickel, 1986) for an overview of theorem proving strategies; it covers the linearity and the unit-resulting restriction). This is not trivial, since although each strategy *alone* is complete for Horn theories, their naive combination is not. Furthermore we insist on completeness for an *arbitrary* goal literals taken from the input set. All these properties are motivated by the intended application within linear theory reasoning calculi. This will be insinuated in Section 1.1 below. For a more detailed treatment we refer the reader to (Baumgartner, 1994).

As an example consider the theory $\mathcal{T}$ of strict orderings which is axiomatized by the clauses

$$\mathcal{T} = \{\neg(x < x), \quad (x < y) \wedge (y < z) \rightarrow (x < z)\}$$

Linearizing completion produces the following finite set $\mathcal{I}_\infty(\mathcal{T})$ of clauses:

$\mathcal{I}_\infty(\mathcal{T})$:

$x < x \rightarrow \mathit{false}$	(Irref)
$x < y \rightarrow \neg(y < x)$	(Asym)
$x < y, y < z \rightarrow x < z$	(Trans-1)
$\neg(x < z), y < z \rightarrow \neg(x < y)$	(Trans-2)
$x < y, \neg(x < y) \rightarrow \mathit{false}$	(Syn)

The associated *operational* meaning of, for instance, the clause (Trans-1) is "from literals $x < y$ and $y < z$ infer the literal $x < z$". Consequently, we call such clauses *inference rules*. Under this operational viewpoint it is clear that (Trans-1) and (Trans-2) are different,

although they are logically equivalent. Now let $\mathcal{L}$ be the $\mathcal{T}$-unsatisfiable input literal set

$$\mathcal{L} = \{\neg(a < b), c < b, a < c\}$$

In order to prove $\mathcal{L}$ as $\mathcal{T}$-unsatisfiable we can chain applications of the inference rules from $\mathcal{I}_\infty(\mathcal{T})$ to $\mathcal{L}$. If the goal literal $\neg(a < b) \in \mathcal{L}$ is chosen we find the following refutation:

$$R_1 = (\neg(a < b) \overset{c<b}{\Longrightarrow}_{Trans-2} \neg(a < c) \overset{a<c}{\Longrightarrow}_{Syn} false)$$

Here, the goal literal $\neg(a < b)$ is transformed stepwise using literals from the input set $\mathcal{L}$ (which are written on top of the arrows) according to the above mentioned operational meaning of inference rules (Section 1.4 explains our notation in more detail).

Recall from above that we demand completeness for an *arbitrary* goal literals. This need stems from the abovementioned intended application of linearizing completion within linear, goal-sensitive calculi, because in such a setting the starting point for a refutation — i.e. the goal — is fixed in advance. Thus, in the example, a refutation with goal literal $a < c \in \mathcal{L}$ should also exist. It is as follows:

$$R_2 = (a < c \overset{c<b}{\Longrightarrow}_{Trans-1} a < b \overset{\neg a<b}{\Longrightarrow}_{Syn} false)$$

The use of the (Trans-1) and of the (Trans-2) inference rules in the refutations R_1 and R_2 should indicate that indeed both of them are needed for completeness.

1.1. Linearizing Completion and Theory Reasoning

As mentioned above the development of linearizing completion was initiated by the desire to automatically construct inference systems for *theory reasoning* calculi. Theory reasoning means to relieve a calculus from explicit reasoning in some domain (e.g. equality, partial orderings) by taking apart the domain knowledge and treating it by special inference rules ((Stickel, 1985; Baumgartner, 1992b; Baumgartner, 1992a; Baumgartner, 1994); (Baumgartner *et al.*, 1992) contains an overview). In an implementation, this results in a universal "foreground" reasoner that calls a specialized "background" reasoner for theory reasoning. For example, in the treatment of mathematical problems, besides equality, other relations such as strict and/or partial orderings are often used ((Bledsoe, 1990) contains challenging problems in this domain).

In order to explain the rôle of linearizing completion, we have to go a little further into the details of theory reasoning. Theory reasoning comes in two variants (Stickel, 1985): *total* and *partial* theory reasoning. *Total* theory reasoning lifts the idea of finding syntactical complementary literals in inferences to a semantic level. Let us briefly describe this general mechanism in the case of *theory model elimination* as described in (Baumgartner, 1994). For this it is helpful if the reader is familiar with the tableaux notation of model elimination ((Baumgartner and Furbach, 1993; Letz *et al.*, 1992)).

As a sample theory let us consider strict orderings, i.e. transitive and irreflexive relations, and assume that the "$<$"-predicate shall be interpreted in this way. Then, for

example, if in a tableau a branch ending in $\neg(a < b)$ is given (Tableaux $\boxed{1}$ in Figure 1a), then theory model elimination might select additional literals $\{c < b, a < c\}$ from input clauses[1] and pass the whole *key set* $\{\neg(a < b), c < b, a < c\}$ to the background reasoner. The background reasoner in turn should discover that this key set is contradictory. Finally, the involved clauses are fanned (in any order) below the selected branch in such a way that a branch containing the key set comes up (cf. Tableaux $\boxed{2}$ in Figure 1a). Since, semantically, a branch is a conjunction of its literals the branch containing the key set is marked with a "$\star$" as solved; the rest literals of the involved clauses (Q and R) constitute new proof obligations.

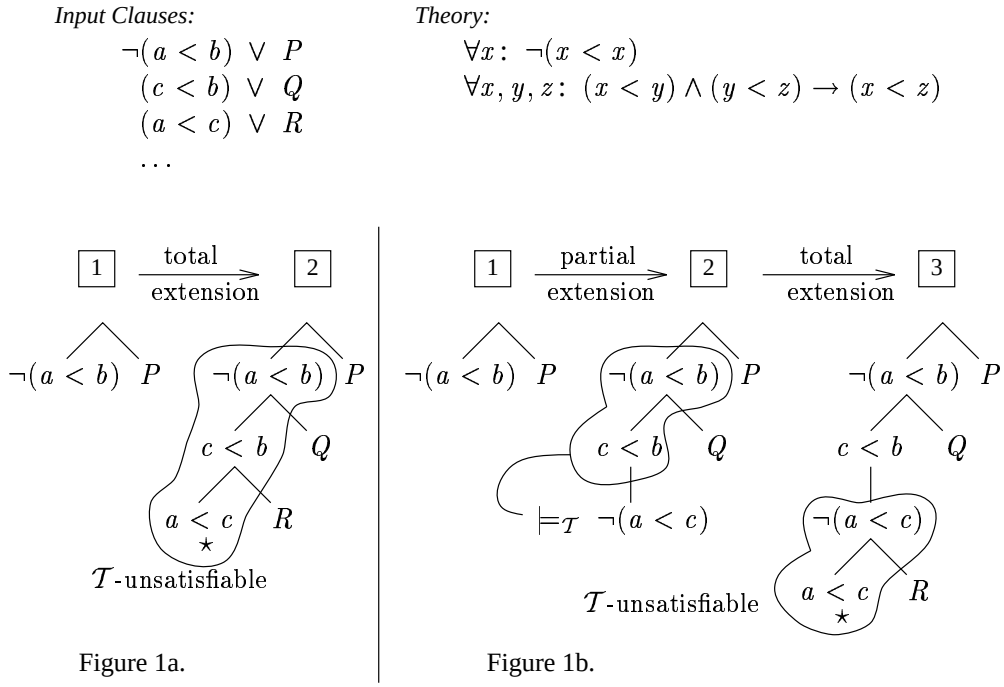

Figure 1a. Figure 1b.

Fig. 1. A *total* theory model elimination derivation (Fig. 1a) and a *partial* theory model elimination derivation (Fig. 1b).

From a practical viewpoint *total* theory reasoning is problematic for some theories. Since in general it cannot be predicted *which* literals constitute a contradictory set, the total inferences might be overly complex, and thus most of the computation would be carried out in the background reasoner. For instance, if the theory is "equality", then total theory reasoning reduces to the *rigid E-unifiability* problem ((Gallier *et al.*, 1987)). Even worse, in general the background reasoner cannot be designed as an always terminating procedure, because the $\mathcal{T}$-complementary problem for the underlying theory might be

[1] Literals from the considered branch may be selected as well.

undecidable. In this case one has to interleave *two* enumeration procedures. For equality reasoning such a procedure (E-resolution) was proposed in (Morris, 1969).

The framework of *partial* theory reasoning offers a solution to these problems. Instead of having to discover a contradictory set in one single "big" total step, the contradictory set is tried to be discovered in a sequence of better manageable, *decidable*, smaller steps. In order to realize this, the result of such a step is stored as a new proof obligation, called the "residue". Hopefully, the residue marks an advance in the computation of the contradictory set.

In the example, the background reasoner might be passed the key set $\{\neg(a < b), c < b\}$, compute the residue $\neg(a < c)$ and return it to the foreground reasoner. The input clauses are used for extending the tableaux much as in the total case, except that instead of closing a branch the residue is added (Tableaux $\boxed{2}$ in Figure 1b).

Clearly, this partial version alone does not suffice since it never closes a branch. Instead the total extension step is allowed as well, however in a very limited way. For instance, in the next step, the foreground reasoner might continue on $\boxed{2}$ by selecting the literal $a < c$ from an input clause and calling the background reasoner with the key set $\{\neg(a < c), a < c\}$. Since this set is complementary the resulting branch can be marked as solved (Tableaux $\boxed{3}$ in Figure 1b).

However, in general, it is not evident *which* key sets for inferences and *which* residues have to be computed along the expansion of a branch. For soundness reasons the residue must be a logical consequence (in some sense) of the passed literals. Clearly, for reasons of search space explosion it is not appropriate to consider *every* such logical consequence as a residue. From this viewpoint, linearizing completion can be understood as a device to compute a search-space restricted background inference system for partial theory reasoning.

Figure 2 depicts the architecture of such a combined system. Note that the background inference system depicted there is just the system $\mathcal{I}_\infty(\mathcal{T})$ described at the end of the previous section. Now, carrying out a *total* extension step wrt. $\mathcal{I}_\infty(\mathcal{T})$ means to restrict to those inferences which can be executed by simultaneously resolving away the premise literals of an inference rule with conclusion $false$ (such as $x < x \rightarrow false$); partial inference are restricted in much the same way, except that the (instantiated) conclusion of the inference rule (unequal to $false$, such as $x < z$ in $x < y,\ y < z \rightarrow x < z$) constitutes the residue.

With our intended strategy of using inference rules to describe extension steps, only "few" possible logical consequences are computed as residues. For instance, in our example theory of strict orderings, we learn from $\mathcal{I}_\infty(\mathcal{T})$ that it suffices to restrict the key set of both partial and total inferences to contain at most *two* literals.

Consequently, we obtain significant efficiency improvements when compared to the naïve approach, where the theories' axioms are supplied as input clauses (Section 9 reports on practical experiments carried out with our theorem prover PROTEIN (Baumgartner and Furbach, 1994b)).

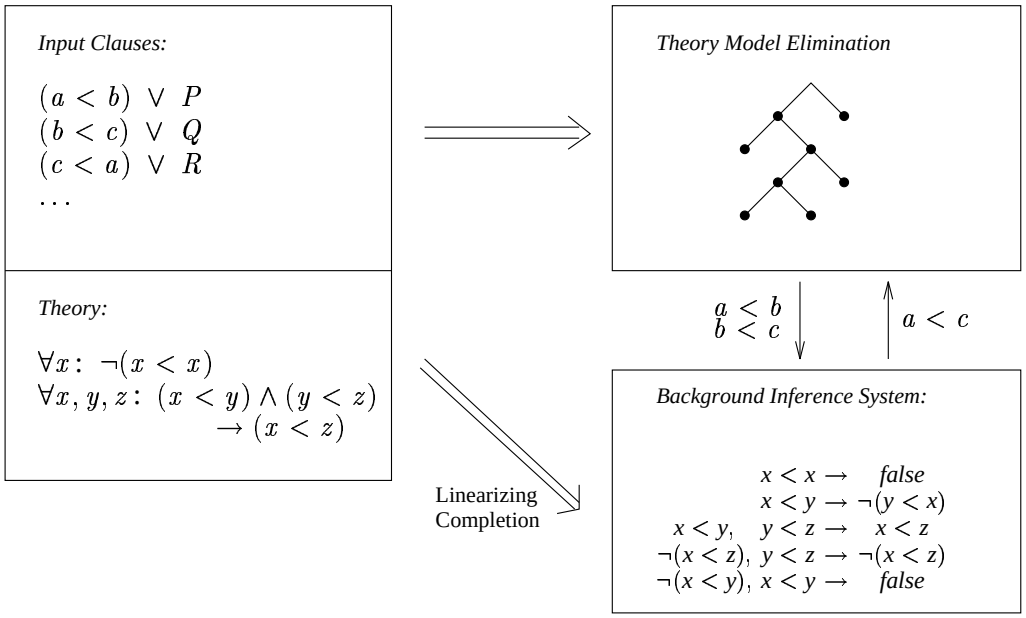

Fig. 2. Application of *linearizing completion* within *partial theory model elimination.*

It remains to link theory model elimination to the formalism of the previous section: when viewing at extension steps in the way just outlined, the refutation R_1 above just describes the theory reasoning necessary to expand the branch in tableaux $\boxed{1}$ (Figure 1b) until it is solved (tableaux $\boxed{3}$); the goal literal of such a refutation corresponds to the leaf literal of the branch to be expanded, and the literals written on top of "$\Rightarrow$" are part of the key sets.

More generally, in (Baumgartner, 1994) it is shown that partial theory model elimination is complete, provided that the background calculus (using a notion of "refutation" as in the present paper) enjoys the strong completeness property, saying that a linear and unit-resulting refutation should exist for any theory-unsatisfiable literal set and arbitrarily selected goal literal[2]. The present paper thus complements the work in (Baumgartner, 1994) in that it shows how such background calculi can be obtained.

The question arises to what extent linearizing completion is tailored towards application within linear, goal-sensitive calculi, such as theory model elimination. Although we have presently not done it, it should be straightforward to use linearizing completion in conjunction with less restricted calculi, such as Stickel's theory resolution. In this case the completeness demand for an *arbitrary* goal literal can be dropped (although it might

[2] This result is shown by breaking all total extension steps of an existing total theory model elimination into a sequence of partial extension steps; this sequence must exist due to the strong completeness property of the background calculus.

be advisable for efficiency reasons to keep it) and the completion requirements can be relaxed a bit. This, however, is not touched by the present paper.

Theory reasoning is related to *constraint reasoning* (See (Bürckert, 1991) for a constraint resolution calculus). On the one hand, constraint reasoning is more general than theory reasoning, as constraints may be treated lazily. Furthermore, no concrete theory-unifiers need to be computed during proof search. Instead it suffices to establish the satisfiability of the accumulated theory unification problems. On the other side, constraint reasoning is more specialized than theory reasoning, as in constraint reasoning the foreground theory must be a conservative extension of the background theory. We are currently designing a constraint model elimination calculus (Stolzenburg and Baumgartner, 1994), and linearizing completion might turn out to be a useful tool in this context, too.

1.2. Related Work

First we will comment on some systems with dedicated theory reasoning components. A system for reasoning with (total) strict orderings was described in (Hines, 1992). It was extended towards an inference rule for set inclusion ($\subseteq$) in (Hines, 1990). In (Bachmair and Ganzinger, 1993) it is demonstrated that the "chaining" inference rule of (Hines, 1992) for transitive relations can be obtained by application of term-rewriting techniques within a more general resolution calculi. Another more general (in the sense that it is not restricted to a single background theory) theory reasoning systems is *Z-Resolution* (Dixon, 1973), which builds in a theory consisting of two-literal clauses only. A more recent improvement was given in (Ohlbach, 1990). Finally, automatizing the theory of *equality* is a research topic of its own. (e.g. "paramodulation" (Robinson and Wos, 1969), or (Bachmair *et al.*, 1992) for a much improved version).

In contrast to our approach, these inference systems are either tailored for one single theory (and thus are not general), or are too restricted (the compilation of two-literal clauses only). As a general method, in (Murray and Rosenthal, 1987) a matrix method with built-in theories is presented. Unlike in Stickel's theory resolution (Stickel, 1985), and similarly to our setting (cf. Section 1.1 above), the theory is *not* considered as a black box. Instead the theory $\mathcal{T}$ is supposed to be defined by a set of clauses. Thus (Murray and Rosenthal, 1987) is even more general than us as the restriction to Horn theories is not necessary. They propose to close $\mathcal{T}$ under application of binary resolution. The resulting — in general infinite — system $\mathcal{T}^*$ is used for total theory extension steps by simultaneously resolving away all literals from a clause from $\mathcal{T}^*$ against given literals in input clauses. The idea of closing the theory under resolution is similar to our completion process. As major differences we have that, first, linearity is not relevant in their context, and, second, they do not apply redundancy very heavily. For instance, only "subsumption" is used as a redundancy criterion. Also, generation of new clauses is not restricted to non-redundant clauses. Altogether, far more clauses will be generated than in our case. See also Section 1.5 below for a more detailed discussion of linearizing completion and resolution.

To our impression, the related work most closest to ours is the approach of the *special relation rules* (Manna and Waldinger, 1986). In the following discussion we will refer to the extension presented in (Manna *et al.*, 1991).

(Extended) special relation rules are inference rules which are derived from certain axiomatically given properties of relations. More specifically, *monotonicity properties* are declarations of properties of relations "$\prec_1$" and "$\prec_2$", which determine the conditions (including polarity) under which subterm replacements can be carried out. Alternatively, these declarations can be described (disregarding polarities) by axioms of the following form:

$$\begin{array}{ll} \mathit{if} \quad u \prec_1 v & \mathit{if} \quad u \prec_1 v \\ \mathit{then} \quad r(\ldots u \ldots) \prec_2 r(\ldots v \ldots) & \mathit{then} \quad r(\ldots v \ldots) \prec_2 r(\ldots u \ldots) \end{array}$$

This scheme is sufficient to cover many interesting relations, such as the axioms (schemes) "symmetry", "transitiviy", "P-substituivity" and "F-substituivity" which comprise — short of "reflexivity" — the axioms of equality. For instance, "symmetry", i.e. $\mathit{if}\ u = v\ \mathit{then}\ v = u$, is immediately seen to be an instance of the right scheme above. "Transitivity" is covered by instantiating in the right scheme "$\prec_2$" with logical implication, "$\rightarrow$". Examples of other theories expressible in this language are ordering relations and subset relations.

It is intended to read these axioms operationally in the way suggested by the notation. The thus derived "special relation (SR) inference rules" are embedded into a resolution calculus, much like our inference rules are embedded into model elimination (see Section 1.1).

The SR inference rules and the inference rules generated from our linearizing completion (LC) compare on the common domain as follows: both are "unit-resulting" (i.e. all premise literals have to be resolved away simultaneously). The SR rules are fewer than the respective LC rules. For instance, no extra contraspositive is required for "transitivity" (cf. $\mathcal{I}_\infty(\mathcal{T})$ above). This is not surprising as no completion takes place. Furthermore, the SR rules are more restrictive than the LC rules, since no replacement below the variable level occurs (cf. Section 6 for a discussion on that).

However, in this comparison it is important to note that the LC rules work in conjunction with a linear, and hence more restricted, calculus than the SR rules. This restriction has large impacts on the completion procedure.

Unlike our LC rules, the SR rules are incomplete (Manna *et al.*, 1991). In the canonical counterexample it would help to replace subterms *below* the variable level, which is forbidden.

The authors of (Manna *et al.*, 1991) first speculated that the situation can be repaired by adding more inference rules, called *relation matching (RM) rules*. These rules generalize the well-known RUE-resolution inference rule (Digricoli and Harrison, 1986) towards general monotonicity properties. Unlike the SR rules (amd our LC rules), the RM rules are no longer unit-resulting, i.e. the conclusion might consist of more than one literal. Furthermore, it is now necessary to deductively close the RM rules (similar as in in our completion). In order to arrive at a finite system, *variable elimination* rules are

used to simplify resolvents. However, these rules are sound only when making very strong assumptions about the underlying relations; this restricts the method's applicability. Finally, as said in (Manna *et al.*, 1991), completeness is still open.

As an overall evaluation, we feel that our paper represents some progress towards solving open issues in the Manna, Stickel and Waldinger paper.

Other sources for related work are the completion techniques developed within the term-rewriting paradigm. In fact, linearizing completion was inspired by Knuth-Bendix completion (Knuth and Bendix, 1970) (cf. also Section 1.3 below) and its successors (e.g. (Hsiang and Rusinowitch, 1987; Bachmair *et al.*, 1986; Bachmair *et al.*, 1989)).

Knuth-Bendix completion has been generalized to conditional equational theories (e.g. (Kaplan, 1987; Dershowitz, 1990; Bachmair, 1991; Dershowitz, 1991a; Dershowitz, 1991b; Ganzinger, 1991)) i.e. definite clauses with built-in equality, and even to full first-order equational theories, e.g. (Bachmair and Ganzinger, 1990; Zhang and Kapur, 1988; Nieuwenhuis and Orejas, 1990; Nieuwenhuis and Rubio, 1992; Bronsard and Reddy, 1992).

These approaches are often more general in a certain respect than ours: they allow for *equational* specifications, whereas we do not have a dedicated treatment for equations[3]. There are several ways to translate Horn logic into equational logic. As a first method, at least for propositional logic, a *formula* F over usual logical connectives can be translated into an equation $F = true$ which then is processed by a term rewriting system for Boolean algebra (Hsiang and Dershowitz, 1983; Paul, 1985; Paul, 1986). It is even possible to model linear input strategies for Horn theories within specialized versions of extended (by associative-commutative operators) Knuth-Bendix completion (see e.g. (Dershowitz, 1985)). However, the unit-resulting restriction and the independence of the goal literal are not considered in those settings.

Another, straightforward, translation of Horn logic into equational clause logic results from simply reading a literal A as the equation $A = true$ (over a different signature). Note that since we allow purely negative clauses such as $\neg(x < x)$ in the specification to be completed, this translation requires full first-order clauses and not just definite clauses. It is common to methods operating on such equational clauses that they rely heavily on *term-orderings* for certain purposes: first, term-orderings are used to select — usually only maximal — literals inside clauses for inferences; second, restricted versions of paramodulation are directed in an order-decreasing way; finally, redundancy is typically defined employing term-orderings.

Here we see a major difference between these techniques and ours, as the *linearity* and the *unit-resulting* restrictions usually are not considered as a restriction for refutations in the completed theory. While we insist on linearity of *derivations*, they consider, more "locally", derivations built from term-ordered *inferences*. Consequently, the notion of a "goal" literal is not a topic of interest.

[3] It is our goal to include equality at a later stage.

An exception is (Bertling, 1990) which describes a procedure to complete towards a combination of term-ordering restrictions and the linear restriction. However, the unit-resulting restriction is not considered there.

As a further difference, for linearizing completion the abovementioned independence of the selection of the goal literal requires the presence of *contrapositives* of the same clause, such as (Trans-1) and (Trans-2) above. In the term-rewriting paradigm this is not necessary.

All these observations indicate to us substantial differences between linearizing completion and the completion techniques described in the term-rewriting literature. To conclude, we cannot see how any of these approaches could be simply instantiated in such a way that linearizing completion results. However, we can take advantage of many standard notions and techniques developed in term-rewriting. This shall be sketched next.

1.3. Relation to Knuth-Bendix completion

As Knuth-Bendix completion (see (Knuth and Bendix, 1970; Bachmair *et al.*, 1986; Bachmair, 1991)), linearizing completion can be understood as a sort of compiler, which compiles a specification once and for all into an efficient algorithm. There are also some analogies in processing: Knuth-Bendix completion relates to equational theories and ordered derivations as linearizing completion relates to general Horn theories and linear derivations. Thus we use a different ordering criteria ("linearity" rather than "orderedness"), and, second, we adopt a more general viewpoint and propose to treat arbitrary Horn theories, not just equality.

Technically, we view a Horn clause $\{\neg L_1, \ldots, \neg L_n, L_{n+1}\}$ as an *inference rule* $L_1, \ldots, L_n \rightarrow L_{n+1}$ which stands operationally for a unit-resulting inference "from $L_1, \ldots, L_n$ infer L_{n+1}". Linearizing completion proceeds by identifying sources for non-linearity in unit-resulting proofs[4] carried out with such inference rules. Non-linearities correspond to "peaks" in the term-rewriting paradigm. In analogy with these peaks, linearizing completion has to invent a new inference rule that repairs the situation. This is done by overlapping inference rules by the so-called **Deduce** transformation operation. Possible nontermination comes in by saturating the rule set under this (and similar) operations. However by the use of redundancy criteria termination is achieved quite often for practically relevant theories. If the completion does not terminate, we arrive at an "unfailing" procedure, i.e. for every provable goal eventually enough inference rules will be generated that are sufficient to prove the goal for such a system. Figure 3 summarizes the relationships between Knuth-Bendix completion and linearizing completion. Some of the concepts listed there will become clear as the text proceeds.

1.4. Informal Description of the Method

Since the main part of this paper is lengthy and quite technical, we prefer to supply a brief and informal description of the method beforehand. Readers familiar with term-rewriting

[4] More precisely: hyperresolution proofs

Concept	*Knuth-Bendix Completion*	*Linearizing Completion*
Underlying Theory	Equality	Arbitrary Horn Theory $\mathcal{T}$
Link Syntax-Semantics	Birkhoff's Theorem	Soundness and Completeness of Unit-Resulting Resolution
Proof Task	$s \stackrel{?}{=} t$	Is $\{K_1, \ldots, K_m\}$ $\mathcal{T}$-unsatisfiable?
Object-Level Inferences	by rewrite rules $u \rightarrow v$	by Inference Rules $L_1, \ldots, L_n \rightarrow L_{n+1}$
Non-normal Form Proof	$s \leftrightarrow^\star t$	Non-linear Unit-Resulting Proof of $\{K_1, \ldots, K_m\}$
Ordering Criteria	Orderedness	Linearity and Unit-Resulting Property
Removal of Peaks in Proofs	By critical pairs turned into rewrite rules	By **Deduce**d inference rules
Normal form Proof	Valley Proof $s \rightarrow^\star u \ ^\star\leftarrow t$	Linear and Unit-Resulting Proof of $\{K_1, \ldots, K_m\}$
First-Order case	Narrowing Proofs	First-Order Derivations
Completeness	Yes: unfailing	Yes: unfailing

Fig. 3. Summary of Relationships between Knuth-Bendix completion and linearizing completion.

will note that in order to express our procedure and our results we have adapted from the term-rewriting paradigm notions like *completion*, *fairness*, *redundancy* and others (see e.g. (Bachmair *et al.*, 1986)) to our needs.

At first we will explain our notion of a linear and unit-resulting proof. The underlying calculus consists of *inference rules*, which are expressions of the form

$$L_1, \ldots, L_n \rightarrow L_{n+1}$$

where all L_is, $1 \leq i \leq n$, are literals (called *premise literals*), and L_{n+1} is either a literal or $false$ (called *conclusion*). The declarative meaning of an inference rule is the implication $\forall((L_1 \wedge \ldots \wedge L_n) \rightarrow L_{n+1})$. Rules with $L_{n+1} \equiv false$ are used to conclude a proof.

As a first step in linearizing completion, a set of inference rules is obtained by rewriting a given Horn theory in an obvious way. Consider e.g. the following theory $\mathcal{O}$ of strict orderings "$<$":

$\neg(x < y) \vee \neg(y < z) \vee x < z$	(Trans)
$\neg(x < x)$	(Irref)

From this theory we construct an *initial* inference system $\mathcal{I}_0(\mathcal{O})$ that contains the following inference rules:

$x < y, y < z \ \rightarrow \ x < z$	(Trans)
$x < x \ \rightarrow \ false$	(Irref)
$x < y, \neg(x < y) \ \rightarrow \ false$	(Syn)

The first two rules stem from the theory immediately, while the additional **Syn**-rule is used to treat syntactical inconsistencies. In general, a given Horn theory S is re-written as an *initial inference system* $\mathcal{I}_0(S)$ (which is a set of inference rules) in the following way: (1) every positive unit clause A becomes $\neg A \rightarrow false$, (2) every definite clause $\neg A_1 \vee \ldots \vee \neg A_n \vee A_{n+1}$ becomes $A_1, \ldots, A_n \rightarrow A_{n+1}$, (3) every completely negative clause $\neg A_1 \vee \ldots \vee \neg A_n$ becomes $A_1, \ldots, A_n \rightarrow false$ and (4) for every predicate symbol p the rule $p(\vec{x}), \neg p(\vec{x}) \rightarrow false$ is added.

Now consider the following literal set:

$$M = \{a < b,\ b < c,\ c < d,\ d < a\}$$

We would like to prove that M is $\mathcal{O}$-unsatisfiable using the inference rules $\mathcal{I}_0(\mathcal{O})$ in a linear and unit-resulting way. The idea of a *linear* proof is to process stepwise an initially chosen goal until a solution is found. We call such initial goals "top literals". Consider $a < b$ as a top literal. Next we have to select an inference rule from $\mathcal{O}$ to be applied to $a < b$. Consider the (Trans) rule for this. The literal $a < b$ can be unified with the premise literal $x < y$ of the (Trans) rule. However, "unit-resulting" means that *all* premise literals of the selected inference rule have to be resolved (using a simultaneous unifier) with some input literals. The input literal $b < c$ can be used for this purpose and the inference rule can be applied then. As a result of this *derivation step* the (instantiated) conclusion of the inference rule is obtained, which is here $a < c$. We write derivation steps more formally as

$$a < b \stackrel{b<c}{\Longrightarrow}_{Trans,\sigma} a < c$$

which means that from $a < b$ using the *side literal* $b < c$, the inference rule Trans and the unifier σ the literal $a < c$ can be derived. We call such steps *linear* because the side literals must be given immediately as a member of the input literal set. Later on we will also temporarily consider *non-linear* derivations, for which the side literals – such as $b < c$ – may also be computed in derivations themselves.

A *refutation* now is simply a chain of derivation steps, terminated by $false$. For the set M the following linear refutation exists (substitutions omitted):

$$a < b \stackrel{b<c}{\Longrightarrow}_{Trans} a < c \stackrel{c<d}{\Longrightarrow}_{Trans} a < d \stackrel{d<a}{\Longrightarrow}_{Trans} a < a \Longrightarrow_{Irref} false$$

Unfortunately, the strategy of simply writing the theory as an initial inference system according to (1)-(4) does not result in a system for which unit-resulting linear resolution is complete. To demonstrate this a different example is needed. Assume the theory consists of the clauses

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

The derived initial inference system then is as follows:

$\mathcal{I}_0(S)$:

$A \rightarrow B$	$A, \neg A \rightarrow false$
$C \rightarrow D$	$B, \neg B \rightarrow false$
$B, D \rightarrow false$	$C, \neg C \rightarrow false$
	$D, \neg D \rightarrow false$

Now let $M_1 = \{A, C\}$. There exists, for instance, a (non-linear) Unit-Resulting refutation of $M_1 \cup S$ (Figure 4) where the Unit-Resulting inferences are carried out as suggested by the arrow-notation of the rules in $\mathcal{I}_0(S)$.

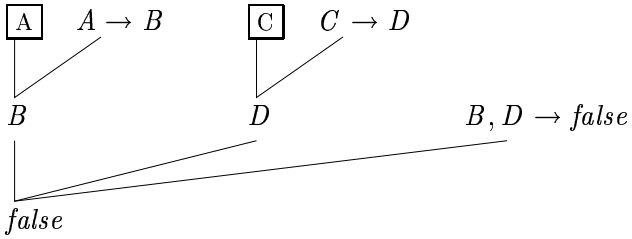

Fig. 4. A non-linear unit-resulting refutation of $\{A, C\} \cup S$, where the inferences are carried out according to $\mathcal{I}_0(S)$. The given input literals — A and C — are boxed.

However, no *linear* refutation of M_1 with top literal A exists (neither does one exist with top literal C for reasons of symmetry); if A is given as the top literal then only the rules $A \rightarrow B$ and $A, \neg A \rightarrow false$ contain the literal A in the premise and thus are the only candidates to be applied. However, the latter is not applicable as $\neg A$ is not given in M_1, and application of the former yields B which also is a dead end (because D is not contained in M_1 and thus $B, D \rightarrow false$ is not applicable). However D could be derived from the input literal C by an application of the rule $C \rightarrow D$. But then, however, due to this auxiliary derivation the refutation would no longer be linear. The same argument holds for the case of C being chosen as top literal.

This problem is solved by linearizing completion by generating a new inference rule that implicitly contains the auxiliary derivation. This generation of new inference rules is the central operation in linearizing completion; it allows a new inference rule to be obtained from two present inference rules by unifying a rule's conclusion with a premise literal of another rule and forming a new rule from the collected premises and the conclusion of the other rule. The new rule then is joined to the present ones. Operations such as this one (and others) on inference systems are described by the device of *transformation rules*. Returning to the last example one can generate a new inference rule by application of the **Deduce** transformation rule in the following way:

$$\frac{\begin{array}{ll} C \rightarrow D & \\ D, & B \rightarrow false \end{array}}{C, \quad B \rightarrow false}$$

Using this new rule $C, B \rightarrow false$ a linear refutation of M_1 can be found. Figure 5 depicts this refutation in the same notation as in Figure 4; in our preferred notation it reads as follows:

$$A \Longrightarrow_{A \rightarrow B} B \overset{C}{\Longrightarrow}_{B,C \rightarrow false} false$$

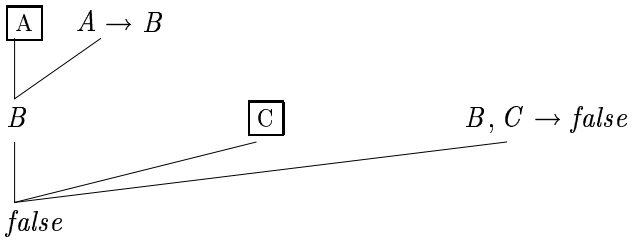

Fig. 5. A linear unit-resulting refutation, using the new rule $B, C \rightarrow false$.

For a first order example of an application of **Deduce** consider again the system $\mathcal{O}$ from above. Two copies of the (Trans) rule can be combined in the following way:

$$\frac{x < y,\ y < z \rightarrow x < z \qquad x' < y' ,\ y' < z' \rightarrow x' < z'}{x' < y,\ y < y' ,\ y' < z' \rightarrow x' < z'}$$

where the unifier for $x < z$ and $x' < y'$ is $\{x \leftarrow x', z \leftarrow y'\}$. In words, the transitivity rule is unfolded once. Repeated application would yield infinitely many unfolded versions of the transitivity rule, but fortunately our redundancy criterion helps to find a finite system here.

This concludes the informal presentation of the **Deduce** transformation rule. Unfortunately, **Deduce** alone does not suffice to obtain completeness as desired. In order to demonstrate the problem here consider the slightly modified example from above:

$$\begin{aligned} S' &= S \cup \{D\} \\ M_2 &= M_1 \cup \{B\} \end{aligned}$$

Of course the old clause $\neg C \vee D$ is not needed for the unsatisfiability of $S' \cup M_2$; but this is not important here. According to the transformation to initial inference systems defined above, the unit clause D becomes the rule $\neg D \rightarrow false$. The initial inference system $\mathcal{I}_0(S')$ looks as follows:

$$\mathcal{I}_1(S): \quad \boxed{\begin{array}{ll} \neg D \rightarrow false & \ldots \\ B, D \rightarrow false & \\ \vdots & \end{array}}$$

Clearly, $S' \cup M_2$ is unsatisfiable, but there is no $\mathcal{I}_0(S')$-refutation of M_2. In particular, the rule $B, D \rightarrow false$ cannot be applied since there is no *input* literal D. On the other hand the literal D is "hidden" in the theory and has been turned into a rule $\neg D \rightarrow false$. In order to obtain completeness it is necessary to consider all such rules of the form $\neg D \rightarrow false$, where D is a positive literal, as an operational substitute for the side literal D in an inference. This could be done either dynamically, i.e. during the proof search, or else in the compilation phase. In order not to spend extra time during the proof search we have decided for the latter alternative. Similar to **Deduce** above, the necessary operations are carried out by transformation rules. In order to solve the example the **Unit2** transformation rule is used, which works in this example as follows:

$$\frac{\begin{array}{rcl} \neg D & \rightarrow & false \\ B, \quad D & \rightarrow & false \end{array}}{\begin{array}{rcl} B & \rightarrow & false \end{array}}$$

Thus, one could say that, when the rules are read as their equivalent clauses, a *unit resolution step* has been carried out. This is what **Unit2** does. There exists a second form, **Unit1**, which is like **Unit2** but for the case when the second involved inference rule contains only one literal in the premise. In both cases the use of a rule $\neg D \rightarrow false$ instead of a side literal D in an inference is simulated. It is easy to see that $\{B\}$ now has a one-step refutation with the new rule $B \rightarrow false$.

All these transformation rules — **Deduce**, **Unit1** and **Unit2** — yield, when applied properly, complete inference systems wrt. the desired linear and unit-resulting restrictions. This is one of our central results. If **Deduce** is omitted, then completeness wrt. the unit-resulting restriction alone results.

These results hold if the transformation rules are carried out in a *fair* way. Fairness means that no possible application is deferred infinitely long. But then the rule generation can be iterated and would result in infinite inference system quite often. For example, the presence of a rule for transitivity alone suffices for infiniteness (the transitivity rule will be unfolded without bound). In order to avoid this, additional transformation rules are needed for the deletion of inference rules. A powerful deletion rule is based on the concept of *redundancy*. Informally, an inference rule is to be redundant if its application in a derivation can be simulated by the other inference rules. As a sufficient and reasonably implementable condition we say that a rule $L_1, \ldots, L_n \rightarrow L_{n+1}$ is redundant in an inference system if there exists a linear derivation of L_{n+1} from input set $L_1, \ldots, L_n$ with any top literal from $\{L_1, \ldots, L_n\}$. For example, the once unfolded version $x' < y, y < y', y' < z' \rightarrow x' < z'$ of the transitivity rule can easily be shown as redundant with this criterion.

Linearizing completion proceeds by repeated fair application of generating and deleting transformation rules to the initial system. Generation can be further restricted: it is fair to generate only new rules from persisting rules, i.e. rules that are generated eventually and never deleted afterwards. Also, only *mandatory* generating transformation rules need to be considered for this (there is also an *optional* transformation rule which allows the addition of a contrapositive of a given rule). Furthermore the result of a generating transformation rule need not be added if it can be shown to be redundant. The result of this process is

a (possibly infinite) system that is closed under derivation of non-redundant inference rules. Such systems are called "complete", as they can be shown to be (refutationally) complete.

There is one thing more to say about deletion: deletion of a redundant inference rule must enable a refutation which is *strictly smaller* wrt. some well-founded ordering on refutations than the refutation which uses the redundant inference rule. This property is crucial for the completion process because it guarantees that an inference system capable of proving a given proof task (provided it is provable at all, of course) will be reached after *finitely* many steps[5]. Concerning implementation this means that we may approximate stepwise infinite inference systems by ever increasing finite systems until a system "large" enough for a concrete proof task is obtained.

We conclude this informal presentation with a note on traditional unit-resulting resolution and on hyperresolution. In traditional unit-resulting resolution (McCharen *et al.*, 1976) *every* literal from a $n+1$ literal clause $L_1 \vee \ldots \vee L_{n+1}$ can serve as the unit resolvent. If this is to be modeled within our inference rules, all $n+1$ contrapositives ($i = 1 \ldots n+1$)

$$\overline{L}_1, \ldots, \overline{L}_{i-1}, \overline{L}_{i+1}, \ldots, \overline{L}_{n+1} \rightarrow L_i$$

have to be used. For example, the transitivity axiom for strict orderings results in the following three contrapositives:

$$\begin{array}{lrcl} 1. & x < y, & y < x & \rightarrow & x < z \\ 2. & \neg x < z, & y < x & \rightarrow & \neg x < y \\ 3. & x < y, & \neg x < z & \rightarrow & \neg y < x \end{array}$$

However, there are cases where not all contrapositives are needed. For example, one of contrapositive 2 or 3 can safely be deleted without affecting completeness. Such restrictions are expressible in our more fine-grained framework, while they are not in traditional unit-resulting resolution framework. This motivated us not to use the traditional formalism but to define a new one.

Hyperresolution (see e.g. (Chang and Lee, 1973a)) is a complete calculus for Horn clause logic. It implements a bottom-up evaluation by starting from the given positive unit clauses and deriving new unit clauses in a unit-resulting way. In our terminology, only the "natural" contrapositives, such as 1, in the last example are needed for this. However, hyperresolution alone does not suggest any completion procedure and yields inherently non-linear refutations. But hyperresolution refutations will serve as a starting point for the completeness proof of linearizing completion.

[5] Thus our approach of proving termination is similar to the approach of *proof orderings* (Bachmair *et al.*, 1986; Bachmair, 1987; Bachmair, 1991) in equational logic. Indeed we use a similar complexity measure, based on multiset orderings. However we found it advantageous to extend the comparison of refutations by additionally considering (optional) *weights* assigned to the used inference rules.

1.5. LINEARIZING COMPLETION AND BINARY RESOLUTION

As mentioned above, the **Unit1** and **Unit2** rules are instances of *unit resolution*. Similarly, the **Deduce** rule works much like the traditional binary resolution inference rule (see e.g. (Chang and Lee, 1973b)). In fact, it is merely a suggestive notation for it which seems appropriate for our purposes. So the question might come up where linearizing completion is different from ordinary resolution.

First, ordinary resolution does not have a restriction to certain contrapositives, as just explained.

Second, every refutation in the linearizing completion paradigm can be simulated stepwise by an ordinary resolution refutation in the following way, but the other direction does not hold. Let $\mathcal{T}_0$ be the Horn clause theory subject to linearizing completion, M be a $\mathcal{T}_0$-unsatisfiable literal set, and $G_0 \in M$ be the desired top literal of the refutation.

Then a refutation in the linearizing completion framework can be written as a resolution refutation

$$\mathcal{T}_0, \mathcal{T}_1, \ldots, \mathcal{T}_n, G_0, G_1, \ldots, G_n, false$$

where (1) the $\mathcal{T}_i$'s are obtained from the $\mathcal{T}_{i-1}$'s by application of binary resolution, corresponding to the transformation rules of linearizing completion, and (2) the G_j's are obtained from the G_{j-1}'s and literals from M by unit-resulting resolution, using a nucleus clause from $\mathcal{T}_n$. These unit-resulting steps could be simulated by sequences of ordinary resolution steps, of course.

It is apparent that this refutation is a highly structured one; evidently not every ordinary resolution refutation is structured in that way. Thus, stepwise simulation in the other direction does not hold.

Third, linearizing completion uses powerful redundancy criteria which are usually not applied in ordinary resolution.

Fourth (connected with three), linearizing completion functions as a compiler, which allows for careful analysis of the input clause set independent of the proof task to be given later. This means that $\mathcal{T}_n$ can be computed once and for all. This is not done to that extent in ordinary resolution.

The rest of this paper is organized as follows: the next section recalls some *preliminaries*. Section 3 introduces *inference systems*; it also contains the completeness of initial inference systems for non-linear refutations. As mentioned in the introduction, deletion of redundant inference rules is tightly coupled with associated orderings of derivations. Section 4 describes our *orderings and redundancy*. Then we are prepared for the *transformation systems* of Section 5. This section introduces the related important notions of *limit* and *fairness* of a deduction, and also that of a *completed* inference system. There it will be shown that the defined transformation rules and redundancy criterion never lose a once found refutation. In Section 6 we will apply the material developed so far to a non-trivial example. In Section 7 we carry on Section 5 and show that our transformation systems have the *complexity-reducing* property, which means that eventually a normal-form refutation will be reached. Then, in Section 8 the material developed so far will be assembled into various *completeness results*; notably, first-order results are

also contained. Section 9 describes some practical experiments carried out for linearizing completion in combination with a theory model elimination theorem prover. Section 10 contains conclusions. Finally, a quite lengthy appendix contains the longer proofs of this paper.

2. Preliminaries

Multisets are like sets, but allow for multiple occurrences of identical elements. Formally, a multiset N over a set X is a function $N : X \mapsto \mathbb{N}$. $N(x)$ is intended as the "element count" of x. We often say that x "occurs $N(x)$ times in N". The multiset union $N \cup M$ of two multisets is described by the equation $(N \cup M)(x) = N(x) + M(x)$, and the multiset difference by $(N - M)(x) = N(x) \dot{-} M(x)$, and the intersection is given by the minimum function. Finally, two multisets are equal, $N = M$ iff $N(x) = M(x)$ for every $x \in X$. We will use set-like notation with braces '$\{\!\{$' and '$\}\!\}$' for multisets. For example the set for which $N(a) = 3$ and $N(b) = 1$ and $N(x) = 0$ for all other values can be written as $\{\!\{a, a, a, b\}\!\}$. If a *set* is used below where a *multi*set were required, then the type conversion is done in the obvious way.

A *multiset with weight over a set X* is a tuple $\langle N, w\rangle$, also written as N_w (or $\{\!\{ \ldots \}\!\}_w$) where N is a multiset over X and $w \in \mathbb{N}$. Thus, a multiset with weight is obtained from an ordinary multiset by attaching some integer to it. As a recursive generalization, define a *nested multiset with weight over a set X* as either an element from X or else as a multiset with weight over a nested multiset with weights. For instance, $\{\!\{a, \{\!\{b, c, \{\!\{d\}\!\}_3\}\!\}_4, e\}\!\}_2$ is such a set over $\{a, b, c, d, e\}$. They will be used as a complexity measure for proofs below.

Furthermore we make heavy use of the data structure 'sequence'. If in the computations below a sequence appears where a multiset is required, the transformation from sequences to multisets is done in the obvious way.

Concerning substitutions we adopt the usual definitions (see e.g. (Lloyd, 1984)). The functions dom and ran denote the domain and range of substitutions, respectively. Sometimes it is necessary to restrict substitutions. Following (Siekmann, 1989) $\sigma|_V$ means the *restriction of the substitution σ to the variable set V*, i.e. $\sigma|_V(x) = \sigma(X)$ for all variables $x \in V$, and is the identity otherwise. By $\sigma < \delta \quad [V]$ we indicate that σ *is more general than* δ *on the domain V* which means that exists a substitution γ such that $\sigma\gamma|_V = \delta|_V$.

Substitutions are applied to multisets and sequences as expected. *Unification* is extended to multisets of literals as follows: A substitution σ is a unifier for N and M iff $N\sigma = M\sigma$. Multiset unification is of type "finitary" (i.e. results in a finite complete set of MGUs). See (Büttner, 1986) for a unification algorithm.

We are mostly interested in Horn theories, which we formalize as follows: A *clause* is a multiset of literals, written as $L_1 \vee \cdots \vee L_k$. A *Horn clause* contains at most one (occurrence of a) positive literal. A *definite clause* contains exactly one positive literal. A *unit clause* contains exactly one literal. A *purely negative* clause contains only negative literals. A *(Herbrand) interpretation* for a given clause set is a subset of the Herbrand base

which does not contain the literal $false$. Intuitively, an interpretation just contains the *true* atoms. More precisely, an interpretation I *satisfies* a ground atom A iff $A \in I$, I satisfies a negative ground literal $\neg A$ iff $A \notin I$, I satisfies an existentially (resp. universally) quantified formula (with $\star \in \{\wedge, \vee\}$) $\exists(L_1 \star \cdots \star L_n)$ (resp. $\forall(L_1 \star \cdots \star L_n)$) of literals iff for some (resp. every) ground substitution γ, I satisfies $(L_1 \star \cdots \star L_n)\gamma$, where "$\wedge$" and "$\vee$" are interpreted as usual. I satisfies a set of ground literals iff I satisfies every literal in it, and I satisfies a ground Horn clause c iff I satisfies some literal in c. An interpretation *satisfies* a Horn clause iff it satisfies every ground instance, and it satisfies a Horn clause set iff it satisfies every clause in it. An interpretation satisfies a literal set iff it satisfies every ground instance of every literal in that set. A Horn clause set is called *satisfiable* iff it is satisfied by some interpretation, or else it is called *unsatisfiable*.

A *Horn theory* $\mathcal{T}$ is a satisfiable set of Horn clauses. A $\mathcal{T}$-interpretation is an interpretation which satisfies $\mathcal{T}$. Let M be a literal set or a clause set. Then M is called *$\mathcal{T}$-satisfiable* iff M is satisfied by some $\mathcal{T}$-interpretation, otherwise M is called *$\mathcal{T}$-unsatisfiable*. It is easily verified that M is $\mathcal{T}$-unsatisfiable iff M is $false$ in every $\mathcal{T}$-interpretation.

3. Inference Systems

Inference systems play the same role as *sets of rewrite rules* in Knuth-Bendix completion: they are used for inferences on the object-level. Inference systems are the objects of computation by *transformation systems* which are introduced in the next section. This section introduces inference systems and basic properties such as "linearity" of proofs.

An *inference rule* (or *rule* for short) is a triple $P \rightarrow_w C$, where P is a multiset of literals, w is a non-negative integer and C is a literal. C may also be the "new" literal $false$, which is assumed to be distinct from all other literals. P is called the *premise*, w is called the *weight* and C is called the *conclusion* of the inference rule. In the sequel we will assume that w is always taken from some finite set $W \subset \mathbb{N}$. The *weights* are motivated by the possibility of extended redundancy checks.

Some notational conveniences: in inference rules we will often write $L_1, \ldots, L_n \rightarrow_w C$ instead of $\{\!|L_1, \ldots, L_n|\!\} \rightarrow_w C$ and $P, Q \rightarrow_w C$ instead of $P \cup Q \rightarrow_w C$, etc. Also, the weight is often dropped if not relevant in the context. The declarative reading of an inference rule $L_1, \ldots, L_n \rightarrow_w C$ is $\forall(L_1 \wedge \cdots \wedge L_n \rightarrow L_{n+1})$.

An *instance* of an inference rule is obtained by application of a substitution to both the premise and the conclusion. *Ground* inference rules do not contain variables. A *(ground) inference system* is a set of (ground) inference rules. If $\mathcal{I}$ is an inference system then $\mathcal{I}^g$ is defined as the inference system consisting of all ground instances of all rules from $\mathcal{I}$.

Next we define how inference shall be used. We say that a literal C' is *inferred from a literal multiset P' by an inference rule $P \rightarrow_w C$ and substitution δ*, written as an *inference*

$$P' \Longrightarrow_{P \rightarrow_w C, \delta} C'$$

iff $P' = P\delta$ and $C\delta = C'$. By overloading of notation, P' is also called the *premise*, C' is called the *conclusion* of the inference and $P \rightarrow_w C$ is called the *used* inference rule. The

inference is called *ground* iff $(P \rightarrow_w C)\delta$ is ground (note that the premise and conclusion of the inference thus are also ground). We will often abbreviate $P' \Longrightarrow_{P \rightarrow_w C, \delta} C'$ to $P' \Longrightarrow_{P \rightarrow_w C} C'$ or even $P' \Longrightarrow C'$ if context allows.

Next we are going to inductively define $\mathcal{I}$*-derivations* based on this notion of inference:

1. The literal L_1 is an $\mathcal{I}$-derivation of L_1 from input literals M with top literal L_1 and length 1. Such a derivation is also called *trivial*.

2. If

   a) D_n is an $\mathcal{I}$-derivation of L_n from input literals M with top literal L_1 and length n, and

   b) $D_n^1, \ldots, D_n^{m_n}$ ($m_n \geq 0$) are $\mathcal{I}$-derivations (called *side derivations* in this context) of *side literals* $L_n^1, \ldots, L_n^{m_n}$, respectively, from M, all respective top literals are contained in M, and

   c) there exists an inference rule $P_n \rightarrow_{w_n} C_n \in \mathcal{I}$, and

   d) there exists a substitution δ_n such that

   $$L_n, L_n^1, \ldots, L_n^{m_n} \Longrightarrow_{P_n \rightarrow_{w_n} C_n, \delta_n} L_{n+1}$$

   then

   $$D_{n+1} = (D_n \xRightarrow{D_n^1 \cdots D_n^{m_n}}_{P_n \rightarrow_{w_n} C_n, \delta_n} L_{n+1})$$

   is a $\mathcal{I}$-derivation of L_{n+1} from input literals M with top literal L_1 and length $n+1$.

   If context allows, the involved inference rule and/or the substitution δ_n shall be omitted.

3. Nothing else is a $\mathcal{I}$-derivation.

Note that *derivations* are nothing but syntactically sugared terms over a respective signature, and thus can be subject to structural induction. In non-trivial derivations the principal (4-ary) construct symbol is "$\Rightarrow$", which takes as arguments the derivation D_n derived so far, the sequence $D_n^1 \cdots D_n^{m_n}$ of side derivations, the inference rule and the substitution involved. We will often omit parenthesis and write

$$D_{n+1} = (L_1 \xRightarrow{D_1^1 \cdots D_1^{m_1}}_{P_1 \rightarrow_{w_1} C_1, \delta_1} L_2 \cdots L_n \xRightarrow{D_n^1 \cdots D_n^{m_n}}_{P_n \rightarrow_{w_n} C_n, \delta_n} L_{n+1}) \quad (1)$$

Occasionally, we will use the symbol ε to denote the empty sequence of derivations.

Some more terminology is convenient: if D is a derivation of L we also say that L is the *derived* literal of D; the derivation D is called *ground* iff every of its inferences is ground; D is called a *refutation* iff D is a derivation of $false$. An inference rule is said to be *used* in D iff (some instance of) it is used in some inference in D. D is called

linear iff every side derivation occurring in it is a trivial derivation. Otherwise D is called *non-linear*. We will write

$$L_1 \Longrightarrow^*_{\mathcal{I},M} L_n$$

to denote the fact that a $\mathcal{I}$-derivation of L_n from M with top literal L_1 exists; similarly the notation

$$D = (L_1 \Longrightarrow^*_{\mathcal{I},M} L_n)$$

means that D is such a derivation.

Note that a derivation does not instantiate the input literals. This is the same as in a "rewriting" proof in the term rewriting paradigm. Carrying on this analogy, a derivation relates to a first-order derivation (to be defined in Section 8) much like "rewriting" relates to "narrowing" (Hullot, 1980).

The top literal plays the role of a goal to be proved. We are interested in arbitrary goal literals, not just negative literals as usually defined for SLD-resolution. Positive goal literals arise naturally: think e.g. of an inference system for strict orderings, containing a rule $X < X \rightarrow \mathit{false}$ for irreflexivity. A provable query then is for example (in Prolog notation) $?- \neg(a < a)$.

3.1. Initial Inference Systems

As the first step of linearizing completion a given Horn theory $\mathcal{T}$ is re-written in a straightforward way as a set of inference rules:

DEFINITION 1. **(Initial Inference Systems)** Let $\mathcal{T}$ be Horn theory and let $W \subset \mathbb{N}$ be a finite set of weights. The *initial inference system of* $\mathcal{T}$, $\mathcal{I}_0(\mathcal{T})$, is the inference system consisting of the rules

1. $\neg A \rightarrow \mathit{false}$ for every positive unit clause $A \in \mathcal{T}$, and
2. $A_1, \ldots, A_k \rightarrow \mathit{false}$ for every purely negative Horn clause $\neg A_1 \vee \ldots \vee \neg A_k \in \mathcal{T}$ ($k \geq 1$), and
3. $A_1, \ldots, A_k \rightarrow B$ for every definite clause $\neg A_1 \vee \ldots \vee \neg A_k \vee B \in \mathcal{T}$ ($k \geq 1$), and
4. $Q(x_1, \ldots, x_n), \neg Q(x_1, \ldots, x_n) \rightarrow \mathit{false}$ for every n-ary predicate symbol Q in $\mathcal{T}$.

Furthermore, some arbitrary chosen weight $w \in W$ is attached to every rule in $\mathcal{I}_0(\mathcal{T})$. Note that with the theory being satisfiable, the empty clause is not contained in $\mathcal{T}$. Thus for every clause in $\mathcal{T}$ exactly one case applies.

EXAMPLE 2. Let $\mathcal{T} = \{\neg A \vee B,\ \neg C \vee D,\ \neg B \vee \neg D,\ D\}$ be a ground Horn theory (it was also given as S' in the introduction). Then an initial inference $\mathcal{I}_0(\mathcal{T})$ system is

$\mathcal{I}_0(T)$:

$A \rightarrow_5 B$	$A, \neg A \rightarrow_1 \mathit{false}$
$C \rightarrow_4 D$	$B, \neg B \rightarrow_1 \mathit{false}$
$B, D \rightarrow_3 \mathit{false}$	$C, \neg C \rightarrow_1 \mathit{false}$
$\neg D \rightarrow_6 \mathit{false}$	$D, \neg D \rightarrow_1 \mathit{false}$

Equivalence:		*Strict order:*	
$x = x$	(Ref=)	$\neg(x < x)$	(IRef<)
$x = y \rightarrow \quad y = x$	(Sym=)		
$x = y,\ y = z \rightarrow \quad x = z$	(Trans=)	$x < y,\ y < z \rightarrow \quad x < z$	(Trans<)
≤-Substitution:		*f-Substitution:*	
$x = x',\ x < y \rightarrow \ x' < y$	(Sub<-1)	$x = x' \rightarrow f(x) = f(x')$	(Sub$f$)
$y = y',\ x < y \rightarrow \ x < y'$	(Sub<-2)		

Fig. 6. The theory $\mathcal{ES}$ of equality and strict orderings with unary function symbol f.

Next let $M_1 = \{A, C\}$. This is a derivation of $false$ from M_1 with top literal A (weights are omitted):

$$D_1 = (A \Longrightarrow_{A \rightarrow B} B \overset{C \Longrightarrow_{C \rightarrow D} D}{\Longrightarrow}_{B, D \rightarrow false} false)$$

D_1 is non-linear, since the second derivation step violates linearity. Figure 4 in the introduction depicts the same derivation in an alternative notation which emphasizes the tree character of derivations.

EXAMPLE 3. As a non-trivial example consider the joint theory $\mathcal{ES}$ of equality and strict orderings (Figure 6). The predicate symbol $=$ is interpreted as an equivalence relation and the predicate symbol $<$ is interpreted as a strict ordering (i.e. as a transitive and irreflexive relation). Furthermore we have included a single function symbol f of arity 1, which gives rise to the substitution axiom $\forall x : x = y \rightarrow f(x) = f(y)$. The extension to a richer signature is straightforward.

The corresponding initial system is given in Figure 7.

3.2. Completeness of Initial Inference Systems

The completeness result (wrt. unit-resulting refutations) of initial inference systems is needed as a starting point for the proof of the completeness result (wrt. unit-resulting and linear refutations) of the generated inference systems. More precisely, the following line of reasoning applies: given a $\mathcal{T}$-unsatisfiable literal set, by ground completeness there exists a — possible nonlinear — refutation in the initial inference system. By repeated generation of new inference rules in a fair way (see the introduction) eventually an inference system is generated that admits a linear refutation. Furthermore, deletion of redundant inference rules does not destroy linearity.

Thus, ground completeness is the initial link between semantics and syntax and plays much the same role as Birkhoff's completeness theorem (see (Huet and Oppen, 1980)) in Knuth-Bendix completion.

Equivalence:				*Strict order:*			
$x = y, \neg(x = y) \rightarrow$		*false*	(Syn=)	$x < y, \neg(x < y) \rightarrow$		*false*	(Syn<)
$\neg(x = x) \rightarrow$		*false*	(Ref)	$x < x \rightarrow$		*false*	(IRef)
$x = y \rightarrow$		$y = x$	(Sym=-1)				
$x = y,$	$y = z \rightarrow$	$x = z$	(Trans=)	$x < y,$	$y < z \rightarrow$	$x < z$	(Trans<)
≤-Substitution:				*f-Substitution:*			
$x = x',$	$x < y \rightarrow$	$x' < y$	(Sub<-1)	$x = x' \rightarrow f(x) = f(x')$			(Sub$f$)
$y = y',$	$x < y \rightarrow$	$x < y'$	(Sub<-2)				

Fig. 7. The inference system $\mathcal{I}_0(\mathcal{ES})$

As explained in the informal presentation in the introduction, initial inference systems constitute an "almost complete" calculus for the underlying theory (cf. the example of the clause set S' there again). As was previously stated, in order to obtain completeness it is necessary to access all positive literals, no matter whether they are contained in the input set or contained in the theory. Regarding this, we will slightly differ from the introduction in that we will not immediately compile away the "hidden" positive literals, such as $\neg D \rightarrow \mathit{false}$. Instead it is more convenient for us to define for an inference system $\mathcal{I}$ the set

$$\mathit{punit}(\mathcal{I}) = \{A \mid \neg A \rightarrow \mathit{false} \in \mathcal{I}\}$$

as the set of *positive unit clauses from* $\mathcal{I}$. For example, $\mathit{punit}(\mathcal{I}_0(\mathcal{ES})) = \{x = x\}$, and in Example 2 we find $\mathit{punit}(\mathcal{I}_0(\mathcal{T})) = \{D\}$.

The set $\mathit{punit}(\mathcal{I})$ shall temporarily be accessible for derivations as additional input literals (later their usage will be compiled away by respective transformation rules). But then it is clear that the present formalism is just a reformulation of traditional Unit-Resulting resolution (which is biased towards our completion application). Taking the soundness and completeness of traditional Unit-Resulting resolution for granted, it is thus not surprising that respective results can be obtained for our formulation. Nevertheless we will state the precise ground completeness result here, since it will be needed below.

LEMMA 4. **(Ground completeness of $\mathcal{I}_0(\mathcal{T})$)** *Let $\mathcal{T}$ be a ground theory (i.e. a theory consisting of ground clauses only) and M be a set of ground literals. If M is $\mathcal{T}$-unsatisfiable then $L \Longrightarrow^*_{\mathcal{I}_0(\mathcal{T}), M \cup \mathit{punit}(\mathcal{I}_0(\mathcal{T}))}$ false for some $L \in (M \cup \mathit{punit}(\mathcal{I}_0(\mathcal{T})))$.*

For instance, it is easily verified that in Example 2 there exists a $\mathcal{I}_0(\mathcal{T})$-refutation of $\{A\} \cup \mathit{punit}(\mathcal{I}_0(\mathcal{T}))$, but there does not exist a $\mathcal{I}_0(\mathcal{T})$-refutation of $\{A\}$ alone.

Proof. By definition of $\mathcal{T}$-unsatisfiability, M is $\mathcal{T}$-unsatisfiable iff $M \cup \mathcal{T}$ is unsatisfiable, where M is considered as a set of unit clauses. Since the unit-clauses of $\mathcal{T}$ can be used as input literals (via $\mathit{punit}(\mathcal{I}_0(\mathcal{T}))$) an existing Hyper-resolution refutation of

$M \cup \mathcal{T}$ can be reflected in our framework. For an explicit proof from scratch see the appendix[6].

We cannot be content with this completeness result, since (1) the literals $punit(\mathcal{I}_0(\mathcal{T}))$ are needed, and (2) a linear derivation may not always exist. For instance, there does not exists a linear refutation of $M_1 = \{A, C\}$ in Example 2, neither with top literal A nor with top literal C.

3.3. SUBDERIVATIONS

Below we will have to access and replace *subderivations* within derivations. For this let D be a $\mathcal{I}$-derivation[7]

$$D = (L_1 \overset{D_1}{\Longrightarrow} L_2 \cdots L_n \overset{D_n}{\Longrightarrow} L_{n+1})$$

It is apparent that

$$D|_{i,j} := (L_i \overset{D_i}{\Longrightarrow} L_{i+1} \cdots L_{j-1} \overset{D_{j-1}}{\Longrightarrow} L_j)$$

for $1 \leq i \leq j \leq n+1$ is a $\mathcal{I}$-derivation of L_j with top literal L_i, called an *immediate subderivation of D*; it is denoted by $D|_{i,j}$. Immediate subderivations with $j = i+1$ are also called *derivation steps*.

Derivations may be concatenated. If E is a $\mathcal{J}$-derivation of the form

$$E = (K_1 \overset{E_1}{\Longrightarrow} K_2 \cdots K_m \overset{E_m}{\Longrightarrow} K_{m+1})$$

and $L_{n+1} = K_1$ then the *concatenation of D and E*, denoted by $D \cdot E$, is defined as

$$D \cdot E = (L_1 \overset{D_1}{\Longrightarrow} L_2 \cdots L_n \overset{D_n}{\Longrightarrow} K_1 \overset{E_1}{\Longrightarrow} K_2 \cdots K_m \overset{E_m}{\Longrightarrow} K_{m+1})$$

Evidently, $D \cdot E$ exists as $\mathcal{I} \cup \mathcal{J}$-derivation from $M \cup N$, where D (resp. E) is a derivation from M (resp. N). It is easily verified that this concatenation is associative. Hence we can omit parenthesis when writing expressions as $D \cdot E \cdot F$.

In order to recursively access subderivations we need the *positions* of derivations. A *position* is a finite string over nonnegative integers and is written in Dewey decimal notation. The empty string is denoted by λ. The *set of positions of D, P(D)* is the smallest set satisfying:

1. $\lambda \in P(D)$.
2. $i.j.r \in P(D)$ if
   a) D is of the form $L_1 \overset{D_1}{\Longrightarrow} L_2 \ldots L_i \overset{D_i}{\Longrightarrow} L_{i+1} \ldots L_n \overset{D_n}{\Longrightarrow} L_{n+1}$ where $1 \leq i \leq n$,

[6] In the sequel the proofs can often be found in the appendix.
[7] Inference rules and substitutions being omitted for brevity; the D_is stand for *sequences* of derivations.

b) and D_i is a sequence of the form $D_i^1 \cdots D_i^{j-1} D_i^j D_i^{j+1} \cdots D_i^{k_i}$ where $1 \leq j \leq k_i$,

c) and $r \in P(D_i^j)$

Building on positions, subderivations can be introduced: let D be a derivation and $p \in P(D)$. The *(occurrence of a) subderivation of D at position p*, $D|_p$ is defined recursively as follows:

$$D|_p = \begin{cases} D & \text{if } p = \lambda \\ D_i^j|_r & \text{if } p = i.j.r \text{ and } D \text{ is of the form} \\ & L_1 \stackrel{D_1}{\Longrightarrow} L_2 \ldots L_i \stackrel{D_i}{\Longrightarrow} L_{i+1} \ldots L_{n-1} \stackrel{D_{n-1}}{\Longrightarrow} L_n \\ & \text{where } D_i = D_i^1 \cdots D_i^j \cdots D_i^{k_i} \end{cases}$$

For instance, take

$$X = (A \stackrel{B \stackrel{C}{\Longrightarrow} D \stackrel{E}{\Longrightarrow} F \quad G \Longrightarrow H}{\Longrightarrow} I)$$

then $X|_{1.1} = (B \stackrel{C}{\Longrightarrow} D \stackrel{E}{\Longrightarrow} F)$, $X|_{1.2} = (G \Longrightarrow H)$ and $X|_{1.1.2.1} = E$.

For ease of notation we abbreviate the selection of an immediate subderivation of a subderivation, i.e. $(D|_p)|_{a,b}$, to $D|_{p,a,b}$. In words, first p is used to locate a subderivation in D and then this subderivation is returned inbetween the indices a and b. For instance $X|_{1.1,2,3} = (D \stackrel{E}{\Longrightarrow} F)$ and $X|_{1.2,1,1} = G$.

Next we turn to replacement of derivations. Suppose D is an $\mathcal{I}$-derivation from M, and suppose E is an $\mathcal{J}$-derivation from N. Furthermore suppose that $D|_{p,a,b}$ and E have the same top literal and derived literal, respectively. Then the sequence which results from replacing $D|_{p,a,b}$ in D by E, written as $D[E]_{p,a,b}$, evidently is an $\mathcal{I} \cup \mathcal{J}$-derivation from $M \cup N$ with same top and derived literal as in D.

For instance,

$$X[D \stackrel{L}{\Longrightarrow} M \stackrel{N}{\Longrightarrow} F]_{1.1,2,3} = (A \stackrel{B \stackrel{C}{\Longrightarrow} D \stackrel{L}{\Longrightarrow} M \stackrel{N}{\Longrightarrow} F \quad G \Longrightarrow H}{\Longrightarrow} I)$$

More formally, replacement is defined as follows:

$$D[E]_{p,a,b} = \begin{cases} D|_{1,a} \cdot E \cdot D|_{b,n}, & \text{if } p = \lambda \\ D|_{1,i} \cdot (L_i \stackrel{D'}{\Longrightarrow}_{P_i \to C_i, \delta_i} L_{i+1}) \cdot D|_{i+1,n} & \text{if } p = i.j.r \\ \text{where } D' = D_i^1 \cdots D_i^{j-1} D_i^j[E]_r D_i^{j+1} \cdots D_i^{k_i} \end{cases}$$

4. Orderings and Redundancy

The transformation systems defined below allow for the deletion of *redundant* inference rules. Redundancy in turn is based on orderings for derivations. Hence these notions have to be introduced first.

4.1. Orderings on Nested Multisets with Weight

First we recall some preliminaries about orderings. In (Dershowitz, 1987) a detailed survey can be found.

A partial strict ordering[8] $\succ$ on a set X (of terms, for our purpose) is called *well-founded* if there is no infinite (endless) sequence $s_1 \succ s_2 \succ \ldots s_n \succ \ldots$ of elements from X. An ordering on terms is said to be *monotonic* iff $s_1 \succ s_2$ implies $f(\ldots s_1 \ldots) \succ f(\ldots s_2 \ldots)$.

As usual we define $s \prec t$ iff $t \succ s$, $s \preceq t$ iff $s \prec t$ or $s = t$ and $s \succeq t$ iff $t \preceq s$. Below we will order terms and integers, in which case $=$ means the respective identity relation (although any equivalence relation can be used in principle), however with one exception: *finite multisets* can be thought of as terms constructed from the variadic constructor symbol "$\{\!\{.\}\!\}$". Henceforth equality of such terms is understood as multiset equality.

Partial orderings can be extended to partial orderings on tuples by comparing their components – as usual – as follows: Assume n sets X_i equipped with n respective partial strict orderings $\succ_i$ as given. Then an n-tuple $s = \langle s_1, \ldots, s_n \rangle$ is *lexicographically greater* than an n-tuple $t = \langle t_1, \ldots, t_n \rangle$, written as $s \succ_{Lex} t$ iff $s_i \succ_i t_i$ for some i $(1 \leq i \leq n)$ while $s_j = t_j$ for all $j < i$.

Similarly, for a given set X of terms with well-founded strict ordering $\succ$ we will with $\gg$ denote the extension of $\succ$ to (finite) multisets over X. $\gg$ is defined as follows:

$$
\begin{aligned}
X = \{\!\{x_1, \ldots, x_m\}\!\} \gg \{\!\{y_1, \ldots, y_n\}\!\} = Y \\
\quad \text{if } x_i \succ y_{j_1}, \ldots, y_{j_k} \text{ and } X - \{\!\{x_i\}\!\} \gg\!\!= Y - \{\!\{y_{j_1}, \ldots, y_{j_k}\}\!\} \\
\quad\quad \text{for some } i,\ 1 \leq i \leq m \text{ and} \\
\quad\quad \text{for some } j_1, \ldots, j_k,\ 1 \leq j_1 \leq \ldots \leq j_k \leq n\ (k \geq 0)
\end{aligned}
$$

Thus, $X \gg Y$ if Y can be obtained from X by successively replacing an element by finitely many (possibly zero) elements strictly smaller in the base relation.

Next we turn to multisets with weight (cf. Section 2). Since multisets with weight are tuples, they can be compared lexicographically. The resulting ordering $\gg_{MW}$ over multisets with weight over a set X of terms then reads as follows:

$$M_v \gg_{MW} N_w \ := \ (M \gg N \text{ or else } (M = N \text{ and } v > w))$$

where M_v, N_w are multisets with weights over X. Thus we first compare the set-components; if these are equal then the weight gives the decision.

[8] a *strict ordering* is a transitive and irreflexive binary relation.

It is well-known that the orderings $\succ_{Lex}$ and $\gg$ are well-founded provided that the orderings upon which they are based are also (see (Dershowitz, 1987) for an overview of orderings). Thus $\gg_{MW}$ is also well-founded.

Orderings on multisets can be generalized to *nested multisets*(Dershowitz and Manna, 1979). For our purposes we have to go a little further and to compare *nested multisets with weight over a set of constants C.* This ordering is nothing but a recursive path ordering with status (see e.g. (Steinbach, 1990)) where the multiset constructor "$\{\!\{.\}\!\}$" is given a multiset status, and the tuple-constructor $\langle N, w\rangle$ (to attach weights to multisets) is given a left-right status. The proof of proposition 7 below will make this more precise.

DEFINITION 5. **(Nested Multisets with Weight)** Let C be a finite set of constants, ordered by the well-founded ordering $\succ$, and let $W \subset \mathbb{N}$ be a finite set of weights. Define the *nested multiset ordering with weights*, $\gg_{NMW}$, recursively as follows:

1. $X = \{\!\{x_1, \ldots, x_m\}\!\}_v \gg_{NMW} \{\!\{y_1, \ldots, y_n\}\!\}_w = Y$
   if $x_i \gtrdot_{NMW} Y$ for some $i = 1 \ldots m$,
   or $X \gg_{MW} Y$ and $v, w \in W$,
   where $\gg_{MW}$ is the extension
   of $\gg_{NMW}$ to multisets with weight.
2. $X = \{\!\{x_1, \ldots, x_m\}\!\}_v \gg_{NMW} y$
   if $y \in C$ and $x_i \gtrdot_{NMW} y$ for some $i = 1 \ldots m$

3. $x \gg_{NMW} y$
   if $x, y \in C$ and $x \succ y$

For termination proofs the *simplification orderings* are of particular interest; a *simplification ordering on a set X of terms* is defined as a monotonic partial strict ordering $\succ$ on X which possesses the *subterm property*, i.e. $f(\ldots s \ldots) \succ s$, and the *deletion property*, i.e. $f(\ldots s \ldots) \succ f(\ldots\ldots)$. The latter property is required only if f is a variadic function symbol (such as the multiset constructor). Various simplification orderings are known, among them the *recursive path ordering (with and without status)*, the *semantic path ordering* and *lexicographic path ordering*. Simplification orderings are interesting for the following reason:

THEOREM 6. *(*Dershowitz, 1982; Dershowitz, 1987*) Any simplification ordering on a set X of terms is a monotonic well-founded ordering on X.*

Fortunately we have:

PROPOSITION 7. *The ordering $\gg_{NMW}$ is a simplification ordering.*

Proof. Nested multisets with weights to be compared by $\gg_{NMW}$ are terms built from the variadic constructor symbol $\{\!\{.\}\!\}$, the 2-ary sequence constructor $\langle .,. \rangle$ and a set C of constants. Furthermore a finite set $W \subset \mathbb{N}$ is given. Now consider the recursive path

ordering with status $>_{RPOS}$ (see e.g. (Steinbach, 1990)), where $\{\!|.|\!\}$ is assigned a multiset status and $\langle .,. \rangle$ is assigned a lexicographical left-to-right status. The precedence $>_{RPOS}$ on function symbols uses the given well-founded ordering $\succ$ on C; the set W is mapped isomorphically (say, by ϕ) to a set of new constants, and $>_{RPOS}$ is extended order-isomorphically wrt. the ordering $>$ on naturals. Finally, the constructor symbols $\{\!|.|\!\}$ and $\langle .,. \rangle$ are given in $>_{RPOS}$ more weight than $\phi(max(W))$. This implies $\{\!|.|\!\} >_{RPOS} \phi(w)$ for every multiset.

With this definition it can be verified by unfolding $>_{RPOS}$ according to the cases of term structure, that $\succ\!\!\!\succ_{NMW}$ satisfies $>_{RPOS}$. In particular, the condition that the constructor symbols $\{\!|.|\!\}$ and $\langle .,. \rangle$ is given in $>_{RPOS}$ more weight than $\phi(max(W))$ implies that we can obtain $M_v >_{RPOS} N_w$ in case N is a true subset of M (because then $M_v >_{RPOS} w$ holds, as is required by $>_{RPOS}$).[9] Thus, with $>_{RPOS}$ being a simplification ordering (see (Steinbach, 1990)), $\succ_{NMW}$ is also.

4.2. Derivation Orderings

In order to be as general as possible we introduce the following notion:

DEFINITION 8. **(Derivation Ordering)** A binary relation $\succ$ on ground derivations is called a *derivation ordering* iff $\succ$ is a well-founded and strict ordering. Now let D be a derivation. A derivation ordering $\succ$ is called *monotonic* iff $D|_{p,i,j} = G$ and $G \succ F$ implies $D \succ D[F]_{p,i,j}$, where F is a derivation which agrees with G on top literal and derived literal.

Next we will design an appropriate derivation ordering for linearizing completion. For this let D be a derivation

$$D = (L_1 \stackrel{D_1}{\Longrightarrow} L_2 \cdots L_n \stackrel{D_n}{\Longrightarrow} L_{n+1}),$$

and define the *complexity of D, compl*(D) as

$$compl(D) := \{\!|0, \langle compl(D_1), w_1 \rangle, \ldots, \langle compl(D_n), w_n \rangle|\!\}$$

where for a sequence $D^1 D^2 \cdots D^n$ of derivations we define

$$compl(D^1 D^2 \cdots D^m) := \bigcup_{i=1\ldots m} compl(D^i) \ .$$

Thus, $compl(D)$ is a multiset whose elements are nested multisets with weights over the set $\{0\}$. $compl(D)$ contains structural information about the derivation: it expresses the shape (when read as a tree) of the derivation encoded as multisets, and occurrences of input literals are mapped to the dummy element 0 at the leaves. Furthermore, the weight

[9] We conjecture that even without the restriction to finite *W* a simplification ordering results. This proof, however, does not go through in that case.

of a used inference rule comes into the complexity measure as the weight of the multiset corresponding to the rule application. For instance, the derivation D_1 in example 2 has the complexity $\{\!|0, \{\!|\}\!|_5, \{\!|0, \{\!|\}\!|_4\}\!|_3\}\!|$.

In order to compare two derivations D_1 and D_2 we attach the artificial weight 0 to them and use the nested multiset ordering with weights. More formally define

$$D_1 \succ_{Lin} D_2 \text{ iff } \langle compl(D_1), 0\rangle \gg_{NMW} \langle compl(D_2), 0\rangle$$

where the base set X is $\{0\}$ and the extended ordering is the empty ordering. Evidently, a derivation D is *linear* (cf. Section 3) iff its complexity $compl(D)$ is of the form

$$\{\!|0, \{\!|0, \ldots, 0|\!\}_{w_1}, \ldots, \{\!|0, \ldots, 0|\!\}_{w_n}|\!\}$$

where the w_is are are the weights of the used inference rules. The ordering $\succ_{Lin}$ is defined in such a way that when weights are neglected smaller derivations are "more linear".

Fortunately we have:

PROPOSITION 9. *The relation $\succ_{Lin}$ is a monotonic derivation ordering.*

4.3. REDUNDANCY

Building on orderings we come to redundancy. In order to be as flexible as possibly the following definition is rather general and includes the notion of redundancy presently used for the linearizing completion as a special case.

DEFINITION 10. **($\succ$-redundancy)** Let $\succ$ be a derivation ordering, and let $\mathcal{I}$ be an inference system. An inference rule $P \rightarrow C$ is called *$\succ$-redundant in $\mathcal{I}$* ($\mathcal{I}$ need not necessarily contain $P \rightarrow C$) iff

1. $P \rightarrow C$ is not of the form $K \rightarrow false$ where K is a literal, and
2. for every inference system $\mathcal{J}$ with $\mathcal{I} \subseteq \mathcal{J}$ and every ground derivation

$$D = (L_1 \Longrightarrow^*_{(\mathcal{J}\cup\{P\rightarrow C\})^g, M} L_n)$$

which uses $P \rightarrow C$ there also exists a ground derivation $D' \prec D$ of the form

$$D' = L_1 \Longrightarrow^*_{(\mathcal{J}\setminus\{P\rightarrow C\})^g, M} L_n \ .$$

$\succ$-redundancy means that any ground derivation in a possibly extended inference system $\mathcal{J}$ using the redundant inference rule can be replaced by a smaller derivation wrt. $\succ$ which uses at most the input literals as given, and this derivation does not use the redundant rule. The strict decreasing property is required since otherwise the redundancy of a rule in a certain inference system would not carry over to the inference system obtained as the limit of the completion process.

An inference rule of the form $K \rightarrow false$ is never redundant due to (1). The motivation for this comes from the use of such rules in the role of input literals in intermediate stages of completion (cf. also the proof of Lemma 4 above). It turns out that the deletion of a rule $K \rightarrow false$ in general does not provide a substitute for this purpose, even if the rule would be redundant according to condition 2.

In general, $\succ$-redundancy is undecidable. However, $\succ$-redundancy plays a vital role in completion procedures. Thus at least a sufficient condition for $\succ$-redundancy should be given in a more constructive way. For linearizing completion we offer the following criteria:

PROPOSITION 11. **(Sufficient $\succ_{Lin}$-redundancy criterion)** *Let $\mathcal{I}$ be an inference system and $P \rightarrow_w C$ be an inference rule. Suppose that for every $L \in P$ there exists a linear $\mathcal{I} \setminus \{P \rightarrow_w C\}$-derivation from P*

$$L \equiv L_1 \overset{D_1}{\Longrightarrow}_{P_1 \rightarrow_{w_1} C_1} L_2 \overset{D_2}{\Longrightarrow}_{P_1 \rightarrow_{w_2} C_1} L_3 \cdots L_{n-1} \overset{D_{n-1}}{\Longrightarrow}_{P_1 \rightarrow_{w_{n-1}} C_1} L_n \equiv C$$

with $n \geq 1$ and such that for $i = 1, \ldots, n-1$ it holds $\langle P \setminus \{\!\{L\}\!\}, w\rangle \not\succ_{MW} \langle D_i, w_i\rangle$ In this comparison the sequence D_i of literals is to be read as a multiset. Then $P \rightarrow_w C$ is $\succ_{Lin}$-redundant in $\mathcal{I}$.

The proof is by induction on the structure of a derivation, thereby making use of the monotonicity property of $\succ_{Lin}$ in the induction step.

Proposition 11 is valuable, since it gives us a more "local" criteria to detect redundancy than Definition 10. Informally, the condition $\langle P \setminus \{\!\{L\}\!\}, w\rangle \not\succ_{MW} \langle D_i, w_i\rangle$ means that the i-th derivation step either uses strictly fewer side literals (the set D_i) than the side literals $P \setminus \{\!\{L\}\!\}$ of a derivation step carried out with $P \rightarrow_w C$, or else, the side literals are the same, but an inference rule of less weight is used. This check has to be done for every $L \in P$ according to the potential applications of the inference rule in a derivation.

EXAMPLE 12. Consider the following ground inference system:

(1) $A, B, C \rightarrow_4 D$
(2) $A, B, C \rightarrow_3 E$
(3) $E, C \rightarrow_5 D$
(4) $C \rightarrow_5 D$

The rule (1) is $\succ_{Lin}$-redundant in this system. Using Proposition 11 this is checked as follows: for A as top literal consider the derivation $A \overset{BC}{\Longrightarrow}_{(1)} D$. It can be replaced by the derivation $A \overset{BC}{\Longrightarrow}_{(2)} E \overset{C}{\Longrightarrow}_{(3)} D$; for the first derivation step we have

$$\{\!\{B, C\}\!\} = (\{\!\{A, B, C\}\!\} \setminus \{\!\{A\}\!\})$$

Hence the weights must be considered, which yields $3 < 4$. Thus the first derivation step satisfies the condition stated in the proposition. For the second step we have $\{\!\{C\}\!\} \subset \{\!\{B, C\}\!\}$, hence the weights do not matter. The case for B as top literal is similar, and for the top literal C rule (4) can be used.

EXAMPLE 13. Consider the rule $R = x < x', x' < y', y' < z' \rightarrow x < z'$ which expresses a once unfolded transitivity rule $Trans = x < y, y < z \rightarrow x < z$. We claim that R is $\succ_{Lin}$-redundant in an inference system which contains $Trans$. Using Proposition 11 this is checked as follows: for $x < x'$ as top literal consider the derivation

$$x < x' \overset{x'<y' \quad y'<z'}{\Longrightarrow}_R x < z'$$

whose side literals are $\{\!\{x' < y', y' < z'\}\!\}$. It can be replaced by the derivation

$$x < x' \overset{x'<y'}{\Longrightarrow}_{Trans} x < y' \overset{y'<z'}{\Longrightarrow}_{Trans} x < z'$$

Furthermore for the first derivation step it holds $\{\!\{x' < y'\}\!\} \subset \{\!\{x' < y', y' < z'\}\!\}$ and for the second step $\{\!\{y' < z'\}\!\} \subset \{\!\{x' < y', y' < z'\}\!\}$. Similarly there exist derivations with top literals $x' < y'$ and $y' < z'$. Weights are not needed for these derivations.

5. Transformation Systems

Transformation systems are the formal device for transformations of the *inference systems* of the previous section. A transformation system transforms an initial inference system in a *fair* way by application of certain *transformation rules* into a *completed state*. Completed inference systems in turn are refutationally complete wrt. the desired linearity and unit-resulting restrictions.

Transformation systems are a rather general device and allow for the construction of a wide range of restricted inference systems. In this paper we concentrate on the instance "linearizing completion".

DEFINITION 14. **(Transformation system)** A *transformation rule* D with *premise* P and *conclusion* C is an expression of the form $\frac{P}{C}$ where P is a multiset of inference rules and C is an inference rule. By *applying* a transformation rule to a multiset of inference rules P' we mean the matching of P to P' by some substitution μ. The *result* of such an application then is $C\mu$. A transformation rule can be labeled as *mandatory* or *optional*. A *transformation system* consists of a set of transformation rules and a well-founded ordering $\succ$ on derivations.

Let $\mathcal{S}$ be a transformation system. The relation $\mathcal{I} \vdash_{\mathcal{S}} \mathcal{I}'$ on inference systems means that $\mathcal{I}'$ is obtained from $\mathcal{I}$ by either (1) adding the result of an application of a transformation rule from $\mathcal{S}$ to variable disjoint variants of rules in $\mathcal{I}$, or else (2) by deleting a $\succ$-redundant inference rule from $\mathcal{I}$. In case (1) the weight of the added inference rule can be chosen arbitrarily. A *$\mathcal{S}$-deduction from an inference system* $\mathcal{I}_0$ is a sequence $\mathcal{I}_0 \vdash_{\mathcal{S}} \mathcal{I}_1 \vdash_{\mathcal{S}} \cdots \vdash_{\mathcal{S}} \mathcal{I}_n \vdash_{\mathcal{S}} \cdots$. Deductions may be of finite or infinite length.

The transformation system Lin consists of transformation rules given in Figure 8. The transformation rules **Unit1, Unit2** and **Deduce** are labeled as mandatory, and **Contra** is labeled as optional. As the derivation ordering we use $\succ_{Lin}$ as defined in Section 4.

Unit1: $\dfrac{L_1 \to C \qquad L_2 \to false}{\overline{C}\sigma \to false}$ { If $\overline{L_1}\sigma = L_2\sigma$ by MGU σ

Unit2: $\dfrac{L_1, P \to C \qquad L_2 \to false}{(P \to C)\sigma}$ { If (1) $P \neq \emptyset$, and (2) $(\overline{L_1} = L_2)\sigma$ by MGU σ

Deduce: $\dfrac{P_1 \to C_1 \qquad L_2, P_2 \to C_2}{(P_1, P_2 \to C_2)\sigma}$ { If (1) $P_2 \neq \emptyset$, and (2) $C_1\sigma = L_2\sigma$ by MGU σ

Contra: $\dfrac{L, P \to C}{\overline{C}, P \to \overline{L}}$ { If $C \neq false$

Fig. 8. The transformation rules of the transformation system *Lin*. Here, *L* and *C* are literals and *P* is a literal multiset.

EXAMPLE 15. From the ground rules $A \to B$ and $\neg A \to false$ the rule $\neg B \to false$ can be obtained by **Unit1**[10]. Consider the inference system $\mathcal{I}_0(\mathcal{T})$ of Example 2 again. From $B, D \to false$ and $\neg D \to false$ we can obtain $B \to false$ by **Unit2**. **Unit1** and **Unit2** share the same purpose: to eliminate in derivations applications of literals from $punit(\mathcal{I})$ (these are initially needed, as stated in Lemma 4). For instance, the set $\{B\}$ now has a one-step refutation in $\mathcal{I}_0(\mathcal{T}) \cup \{B \to false\}$. From the rules $C \to D$ and $B, D \to false$ in Example 2 the rule $B, C \to_{10} false$ can be **Deduce**d. Deduced rules are used to turn a refutation stepwise into a "more linear" refutation. Again, using $\mathcal{I}_0(\mathcal{T}) \cup \{B, C \to_{10} false\}$ the refutation D_1 from Example 2 of $M_1 = \{A, C\}$ can be linearized with the new rule. We have:

$$D_1' = (A \Longrightarrow_{A \to B} B \overset{C}{\Longrightarrow}_{B,C \to false} false)$$

The complexity of D_1' is $\{\!\{0, \{\!\{\}\!\}_5, \{\!\{0\}\!\}_{10}\}\!\}$ and it can be verified that $D_1 \succ_{Lin} D_1'$.

From the transitivity rule $x < y, y < z \to x < z$ and a copy $x' < y', y' < z' \to x' < z'$ by **Deduce** with $\sigma = \{y \leftarrow x', z \leftarrow z'\}$ one obtains $x < x', x' < y', y' < z' \to x < z'$.

From $A, B \to C$ by **Contra** $\neg C, B \to \neg A$ can be obtained.

[10] Weights are not considered in this example.

This **Contra** transformation rule is an optional rule and thus is not labeled as *mandatory*. It is sometimes valuable in order to come to a finite system (Section 6 contains an example for such a case). However, the **Contra** rule should be applied carefully since it increases the search space of the generated inference systems; applying **Contra** exhaustively will produce every possible contrapositive of a theory clause. This is clearly not intended. Surprisingly, an application of the **Contra** rule may increase the deductive power of an inference system: consider e.g. $\mathcal{I} = \{A \rightarrow B\}$. The only non-trivial $\mathcal{I}$-derivation is $A \Longrightarrow B$. However, when $\mathcal{I}$ is enriched with the contrapositive $\neg B \rightarrow \neg A$, the derivation $\neg B \Longrightarrow \neg A$ also exists. However its application does not increase the refutational power of inference systems.

We collect for later use the following lemma, which states that redundancy persists along transformation steps:

LEMMA 16. *Let $\mathcal{S}$ be a transformation system with derivation ordering $\succ$. Suppose $P \rightarrow C$ is $\succ$-redundant in some $\mathcal{I}$, and suppose that $\mathcal{I} \vdash_{\mathcal{S}} \mathcal{J}$. Then $P \rightarrow C$ is also $\succ$-redundant in $\mathcal{J}$.*

Proof. If $\mathcal{J}$ is obtained by adding a new rule to $\mathcal{I}$ the lemma is trivial by the property $\mathcal{J} \supseteq \mathcal{I}$ in the definition of redundancy (Def. 10). If $\mathcal{J}$ is obtained from $\mathcal{I}$ by deletion of a redundant rule then well-founded induction is used to eliminate every use of $P \rightarrow C$ and the deleted rule in derivations. See the appendix for the full proof.

We conclude this section with a note on soundness. In order to achieve the soundness of the overall approach, one has first to guarantee that all derivations obtainable from initial inference systems are sound. This, however, is clear since they are nothing but Unit-Resulting refutations. Second, one has to guarantee that derivations obtainable from transformed inference systems are sound. The key to this result is the observation that newly generated inference rules are just resolvents of present rules, and hence are *logical consequences* of these.

5.1. Limit Inference Systems

The process of applying the transformation rules of a transformation system may terminate or not. In order to treat both cases in a uniform way it is useful to define the *limit* inference system $\mathcal{I}_\infty$ which is finite if the transformation system eventually does not produce new inference rules any more and infinite otherwise.

DEFINITION 17. **(Limit, (**Bachmair, 1991**))** The *limit of a deduction* $\mathcal{I}_0 \vdash \mathcal{I}_1 \vdash \cdots \vdash \mathcal{I}_n \cdots$ is defined as

$$\mathcal{I}_\infty := \bigcup_i \bigcap_{j \geq i} \mathcal{I}_j$$

The elements of $\mathcal{I}_\infty$ are also called the *persisting* inference rules.

The limit $\mathcal{I}_\infty$ of a deduction is the set of inference rules generated eventually and never deleted afterwards:

LEMMA 18. *Let $\mathcal{I}_0 \vdash \mathcal{I}_1 \vdash \ldots$ be a deduction and suppose $(P \to C) \in \mathcal{I}_\infty$. Then for some k, and for all $i \geq k$: $(P \to C) \in \mathcal{I}_i$*

Proof. By contradiction. Assume $P \to C \in \mathcal{I}_\infty$ and suppose that for all k exists an $i \geq k$ such that $P \to C \notin \mathcal{I}_i$. Hence whenever $P \to C \in \mathcal{I}_k$ then $P \to C \notin \bigcap_{i \geq k} \mathcal{I}_i$. Thus $P \to C \notin \bigcup_k \bigcap_{i \geq k} \mathcal{I}_i$ and hence by definition of $\mathcal{I}_\infty$, $P \to C \notin \mathcal{I}_\infty$. This however contradicts the assumption of the lemma.

This result can be extended for derivations:

PROPOSITION 19. *Let $\mathcal{I}_0 \vdash \mathcal{I}_1 \vdash \ldots$ be a deduction and let D be a $\mathcal{I}^g_\infty$-derivation (resp. $\mathcal{I}_\infty$ derivation). Then for some k, D is also a $\mathcal{I}^g_k$-derivation (resp. $\mathcal{I}_k$ derivation).*

Proof. Let $\{r_1, \ldots, r_n\} \subseteq \mathcal{I}_\infty$ be the (finite) inference system used in the given derivation. For every r_j $(j = 1 \ldots n)$ by Lemma 18 it holds that for some k_j, all $i \geq k_j$: $r_j \in \mathcal{I}_i$. Now take $k := max(\{k_1, \ldots, k_n\})$ and observe that $\{r_1, \ldots, r_n\} \subseteq \mathcal{I}_k$.

The next lemma extends Lemma 16 to the limit $\mathcal{I}_\infty$, i.e. redundant inference rules remain redundant in the limit:

LEMMA 20. *Let $\mathcal{S}$ be a transformation system with derivation ordering $\succ$. Let $\mathcal{I}_0 \vdash \mathcal{I}_1 \vdash \cdots$ be a $\mathcal{S}$-deduction. If for some k, $P \to C$ is $\succ$-redundant in $\mathcal{I}_k$ then $P \to C$ is $\succ$-redundant in $\mathcal{I}_\infty$.*

Also the $punit$-literals persist:

LEMMA 21. *Let $\mathcal{I}_0 \vdash \mathcal{I}_1 \vdash \cdots$ be a deduction. If $L \in$ punit$(\mathcal{I}_k)^g$ then also $L \in$ punit$(\mathcal{I}_\infty)^g$.*

Proof. L is a ground instance of some L', where $\overline{L'} \to false \in \mathcal{I}_k$. By definition of $\succ$-redundancy, a rule $\overline{L'} \to false \in \mathcal{I}_k$ is never $\succ$-redundant and thus is never deleted. Hence $\overline{L'} \to false \in \mathcal{I}_i$ for every $i \geq k$. Thus

$$(\overline{L'} \to false) \bigcap_{i \geq k} \mathcal{I}_i \subseteq \bigcup_{k \geq 0} \bigcap_{i \geq k} \mathcal{I}_i = \mathcal{I}_\infty$$

By this the claim follows.

Lemma 20 and Lemma 21 imply the following central property:

LEMMA 22. *Let $\mathcal{S}$ be a transformation system with derivation ordering $\succ$. Let $\mathcal{I}_0 \vdash \mathcal{I}_1 \vdash \cdots$ be a $\mathcal{S}$-deduction. If there exists a derivation $D = (L \Longrightarrow^*_{\mathcal{I}^g_k, M \cup punit(\mathcal{I}^g_k)} L')$ then there also exists a derivation $D' = (L \Longrightarrow^*_{\mathcal{I}^g_\infty, M \cup punit(\mathcal{I}^g_\infty)} L')$ with $D' \preceq D$.*

5.2. Fairness and Completion

Deductions must be *fair*, which roughly means that no application of a mandatory transformation rule is deferred infinitely long. Fairness is important since it entails that "enough"

inference rules to obtain normal derivations are generated. Our definition of fairness is an adaption of standard definitions in the term-rewriting literature (see e.g. (Bachmair, 1991)).

DEFINITION 23. **(Fairness)** Let $\mathcal{S}$ be a transformation system with derivation ordering $\succ$. A $\mathcal{S}$-deduction $\mathcal{I}_0 \vdash \mathcal{I}_1 \vdash \cdots \vdash \mathcal{I}_n \cdots$ is called *fair* iff whenever $\mathcal{I}_\infty \vdash \mathcal{I}_\infty \cup \{P \rightarrow C\}$ for some application of a mandatory transformation rule from $\mathcal{S}$, then for some k, $P \rightarrow C \in \mathcal{I}_k$ up to renaming, or $P \rightarrow C$ is $\succ$-redundant in $\mathcal{I}_k$.

Fairness states that it is sufficient either to generate an inference rule or to prove it redundant from persisting inference rules only. This notion of fairness enables the use of a "delete as many inference rules as possible" strategy in implementations, since a rule once shown to be redundant is redundant in all subsequent stages (Lemma 20) and thus need not persist.

The next central concept is *completion* which means that only redundant new inference rules can be generated from an inference system. Completion is a useful concept since it allows to characterize refutationally complete inference systems, which is a semantical concept, in a more syntactical way[11].

DEFINITION 24. **(Completion)** Let $\mathcal{S}$ be a transformation system with derivation ordering $\succ$. An inference system $\mathcal{I}$ is *completed (wrt. $\mathcal{S}$)* iff whenever $\mathcal{I} \vdash_{\mathcal{S}} \mathcal{I} \cup \{P \rightarrow C\}$ by application of a mandatory transformation rule from $\mathcal{S}$ then $P \rightarrow C \in \mathcal{I}$ up to renaming or $P \rightarrow C$ is $\succ$-redundant in $\mathcal{I}$.

Fairness, deductions and completion relate as follows:

THEOREM 25. *Let $\mathcal{S}$ be a transformation system and $\mathcal{I}_0$ be an inference system. The limit $\mathcal{I}_\infty$ of a fair $\mathcal{S}$-deduction $\mathcal{I}_0 \vdash \mathcal{I}_1 \vdash \cdots$ is completed wrt. $\mathcal{S}$.*

Proof. By contradiction. Suppose $\mathcal{I}_\infty$ is not completed. Then, for some application of a mandatory transformation rule from $\mathcal{L}$ we have $\mathcal{I}_\infty \vdash \mathcal{I}_\infty \cup \{P \rightarrow C\}$ such that $P \rightarrow C \notin \mathcal{I}_\infty$ and $P \rightarrow C$ is not $\succ$-redundant in $\mathcal{I}$ (*). However, by fairness, for some k, $P \rightarrow C \in \mathcal{I}_k$ or $P \rightarrow C$ is redundant in $\mathcal{I}_k$. This suggests the following case analysis:

Case 1: Suppose $P \rightarrow C \in \mathcal{I}_k$. If $P \rightarrow C$ is not deleted afterwards, i.e. for all $i \geq k$, $P \rightarrow C \in \mathcal{I}_i$, then $P \rightarrow C \in \bigcap_{i \geq k} \mathcal{I}_i$ and thus also $P \rightarrow C \in \mathcal{I}_\infty$. Contradiction to (*). Otherwise, if $P \rightarrow C$ is deleted in some $\mathcal{I}_j$, $P \rightarrow C$ must have been $\succ$-redundant in $\mathcal{I}_{j-1}$. But then by Lemma 20 $P \rightarrow C$ is $\succ$-redundant in $\mathcal{I}_\infty$. Contradiction to (*).

Case 2: If $P \rightarrow C$ is $\succ$-redundant in $\mathcal{I}_k$ then by Lemma 20, $P \rightarrow C$ is $\succ$-redundant in $\mathcal{I}_\infty$. Contradiction to (*).

[11] Note that the term "complete" has two distinct technical meanings here, and the refutational completeness of a completed system depends additionally on how the inference rules are used (unit-resulting linear hyperresolution, in this case), and what deduction rules are used to complete the system.

6. Example: Equality plus Orderings

The purpose of this section is to demonstrate the application of linearizing completion to a non-trivial example. We start with the theory of equality alone and then extend it with strict orderings.

Consider the theory of equality in a language without function symbols and a 2-ary predicate symbol P (Figure 9).

Equivalence:				*P-Substitution:*			
		$x = x$	(Ref=)	$P(x,y)$,	$x = x' \rightarrow$	$P(x',y)$	(Sub$P$-1)
	$x = y \rightarrow$	$y = x$	(Sym=)	$P(x,y)$,	$y = y' \rightarrow$	$P(x,y')$	(Sub$P$-2)
$x = y$,	$y = z \rightarrow$	$x = z$	(Trans=)				

Fig. 9. The theory of equality in a language with a 2-ary predicate symbol P.

The inference system in Figure 10 corresponds to that theory and it is completed wrt. the transformation system Lin (Figure 8). The notation $x \doteq y$ is a nondeterministic notation for $x = y$ or $y = x$. If part of an inference rule, the rule has to be expanded to both cases. This system was obtained as the result of a fair deduction, starting from the

Equivalence:				*P-Substitution:*			
$x = y, \neg(x = y) \rightarrow$		*false*	(Syn=)	$P(x,y), \neg P(x,y) \rightarrow$		*false*	(Syn$P$)
	$\neg(x = x) \rightarrow$	*false*	(Ref=)	$P(x,y)$,	$x \doteq x' \rightarrow$	$P(x',y)$	(Sub$P$-1-1)
	$x = y \rightarrow$	$y = x$	(Sym=-1)	$P(x,y)$,	$y \doteq y' \rightarrow$	$P(x,y')$	(Sub$P$-2-1)
$x = y$,	$y = z \rightarrow$	$x = z$	(Trans=-1)	$\neg P(x,y)$,	$x \doteq x' \rightarrow$	$\neg P(x',y)$	(Sub$P$-1-2)
$\neg(x = z)$,	$y \doteq z \rightarrow$	$\neg(x = y)$	(Trans=-2)	$\neg P(x,y)$,	$y \doteq y' \rightarrow$	$\neg P(x,y')$	(Sub$P$-2-2)

Fig. 10. A completed inference system.

initial inference system (Definition 1) associated with the theory. There, we gave a weight of 0 to every rule. The system in Figure 10 then was obtained semi-automatically with the assistance of our implementation (see Section 9 for a more detailed description) in the following way: first we added the following contrapositives manually by application of the **Contra** transformation rule:

$\neg(x = y), \quad y' = y \rightarrow \neg(x = y')$	Contrapositive of (Trans=)
$\neg P(x,y), \quad x' = x \rightarrow \neg P(x',y)$	Contrapositive of (Sub$P$-1)
$\neg P(x,y), \quad y' = y \rightarrow \neg P(x,y')$	Contrapositive of (Sub$P$-1)

All these rules were given the weight 0. Then the mandatory transformation rules of Lin, i.e. **Deduce**, **Unit1** and **Unit2** were applied repeatedly in an exhaustive way, until

no more non-redundant inference rules could be generated. This part of the construction was carried out fully automatically. The generated rules were given weights of somewhat above 0. Alternatively, we can run the linearizing completion tool in a fully automatic way. Then the same system results as in Figure 10 except that additionally the rule $\neg(x = y) \rightarrow \neg(y = x)$ is generated as a contrapositive of the (Sym=-1) rule. This happens due to a particular heuristic built into the completion procedure, which builds a contrapositive rule if it can successfully be applied in the redundancy proofs of two or more other inference rules.

This system also serves as an example for a useful application of the **Contra** rule, which lies in the proof of redundancy: applying **Deduce** to $x = y, y = z \rightarrow x = z$ and $x = z, \neg(x = z) \rightarrow false$ results in $x = y, y = z, \neg(x = z) \rightarrow false$. In order to avoid an infinite chain of applications of **Deduce** we would like to show $x = y, y = z, \neg(x = z) \rightarrow false$ redundant. For the crucial case with $\neg(x = z)$ as top literal this is shown by the derivation

$$\neg(x = z) \xRightarrow{y=z} \text{(Trans=-2)}\ \neg(x = y) \xRightarrow{x=y} \text{(Syn=)}\ false$$

which uses the (Trans=-2) inference rule (which was supplied manually).

It has been widely studied in the literature how to efficiently mechanize the equality relation. It may thus be interesting how the inference system in Figure 10 relates to well-known approaches.

The most prominent approach to deal with equality is certainly *paramodulation* (Robinson and Wos, 1969) and its refinements. Briefly, the inference system in figure 10 reflects the linear paramodulation calculus for equational theories (see e.g. (Furbach *et al.*, 1989)) without function symbols.

Linear paramodulation (the predicate symbol P shall not be considered) proceeds by repeated subterm replacement in a given goal equation $\neg(s = t)$ until a trivial goal of the form $\neg(s = s)$ has been reached. In order to obtain a corresponding refutation in our system, we have to start with the top literal $\neg(s = t)$. Subterm replacement in paramodulation is mirrored in our system by the (Trans=-2) inference rule, while the derivation of the trivial goal in paramodulation is mirrored by an application of the (Ref=) inference rule.

Note that according to the (Trans=-2) inference rule it is sufficient to paramodulate into the right hand side of a negative equation. Technically this is realized by the absence of certain contrapositives of the (Trans-=) inference rule.

When used as a rule of inference within a linear calculus such as model elimination, paramodulation must also be defined for *positive* goal literals (see (Loveland, 1978)). This is already achieved in our system.

Things become more complicated when function symbols are involved. For the sake of simplicity assume a single 2-ary function symbol f as given. This implies for the theory of equality additional substitution axioms:

$$\begin{aligned} x = x' &\rightarrow f(x, y) = f(x', y) \\ y = y' &\rightarrow f(x, y) = f(x, y') \end{aligned}$$

The thus enhanced theory then can be completed in a way similar to above. The resulting system is *in*finite and contains rules like

$$x = x',\ \neg(f(f(x,y),z) = w) \ \rightarrow\ \neg(f(f(x',y),z) = w) \qquad \text{(Inst-Par)}$$

In general, such rules for subterm replacement are generated for arbitrary depth. These rules have also a counterpart in linear paramodulation: it is well-known (see e.g. (Hölldobler, 1989)) that in linear paramodulation the *functional reflexive axioms, i.e. axioms of the form* $f(x,y) = f(x,y)$ are necessary for completeness. Equivalently, the additional inference rule *instantiation* can be used instead (see again (Hölldobler, 1989)).

If derivations with our inference systems are lifted to first-order derivations (Section 8.3), it is straightforward to show that an application of an instantiation inference rule, followed by a paramodulation step, has the same result as carrying out a first-order inference with a respective inference rule such as (Inst-Par).

These considerations about paramodulation lead us to the following conclusion: linearizing completion did not discover an essentially new calculus for equality treatment. However, it succeeded to re-invent a well-known and fairly efficient calculus, namely linear paramodulation. Furthermore, this was done in an automatic way. As an instance of a general completeness result being proved in subsequent sections (in particular Theorem 44), we thus obtain the completeness of linear paramodulation.

Carrying on this result we will expect numerous useful inference systems to be developed in the future. As a generalization of the above system for equality consider example 3 again. Applying a fair Lin-deduction to the initial system $\mathcal{I}_0(\mathcal{ES})$ results in the (infinite) completed inference system depicted in Figure 11. Note that not all contrapositives of the (Trans$<$) axiom have to be generated. We think that this inference system generalizes the above inference rules for equality in a nice and useful way. Finite approximations of this system can be obtained in a fully automatic way by our implementation.

We conclude with a note on RUE-resolution (Digricoli and Harrison, 1986). Since our approach is successful on rediscovering paramodulation, the question arises whether the inference rules of RUE-resolution could be derived as well. It seems like this is not possible, mainly because in RUE resolution the conclusion of the inference rules are *clauses* but not single literals. However, this unit-resulting property is central to our approach.

7. Complexity-Reducing Transformation Systems

Up to now we have seen that the inference systems generated along a deduction never increase the complexity of a once obtained derivation. However, in order to obtain completeness wrt. normal-form derivations (linear derivations) more is required: the transformation system has to eventually generate inference rules that will strictly *de*crease the complexity of a non-normal derivation.

DEFINITION 26. **(Normal Derivations, Order-Normalizing Transformation System)** A set $\mathcal{N}$ of ground derivations is called *normal* iff $\mathcal{N}$ is downward closed wrt. a given

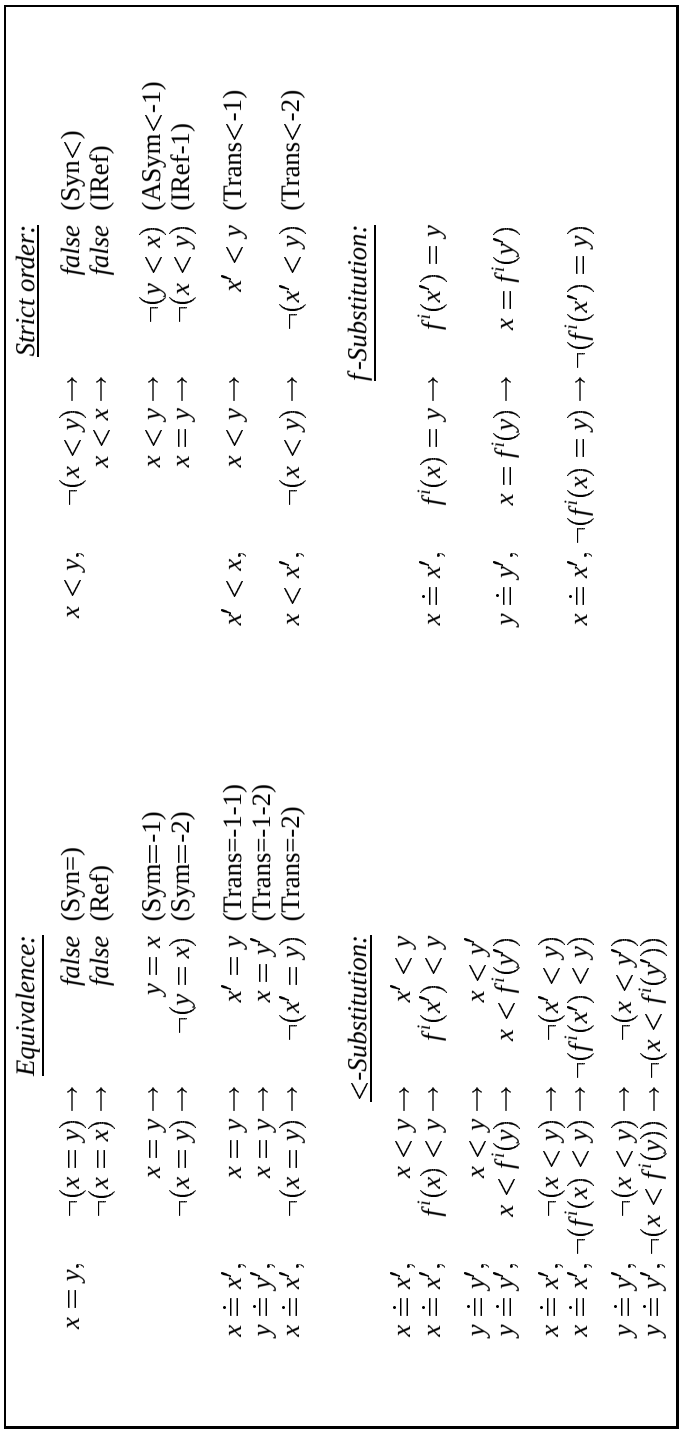

Fig. 11. A completed inference system for the theory $\mathcal{ES}$. Inference rules containing $f^i(x)$ have to be replaced by all i-fold instances $f(f(\cdots f(x)))$, $i > 0$.

derivation ordering $\succ$, i.e. if $D \in \mathcal{N}$ and $D' \prec D$ for some ground derivation D' then $D' \in \mathcal{N}$. In the sequel $\mathcal{N}$ always denotes a set of normal derivations. As an example of a normal set of derivations define the set $LinG$ as the set of all linear ground derivations.

Now let $\mathcal{I}$ be an inference system and let $\mathcal{S}$ be a transformation system with derivation ordering $\succ$. Then $\mathcal{S}$ is called *order-normalizing wrt. $\mathcal{N}$* iff whenever there exists a ground derivation $D = (L \Longrightarrow^*_{\mathcal{I}^g,M} L')$ such that $D \notin \mathcal{N}$ then there exists a derivation $D' = (L \Longrightarrow^*_{(\mathcal{I}')^g,M} L')$ with $D' \prec D$, where $\mathcal{I}' = \mathcal{I}$ or $\mathcal{I} \vdash_{\mathcal{S}} \mathcal{I}'$ by one single application of some mandatory transformation rule.

In words, an order-normalizing transformation system allows generation of a new inference rule (if not present already) that allows a decrease in complexity of a given non-normal ground derivation. Note that not necessarily $D' \in \mathcal{N}$. This property is not required, because the derivation ordering $\succ$ is well-founded. So we will eventually end up with a normal derivation $D' \in \mathcal{N}$ for any given derivation D (see Proposition 28 below).

As an example for an order-normalizing transformation system take "linearizing completion", which is order-normalizing wrt. linear ground derivations:

PROPOSITION 27. *The transformation system Lin is order-normalizing wrt. LinG.*

In essence, the proof uses the facts that the **Deduce** transformation rule is labeled as mandatory and furthermore works towards strictly decreasing derivations wrt. the well-founded ordering $\succ_{Lin}$.

Returning to the general level, we find that a completed inference system in conjunction with order-normalizing transformation systems yields normal derivations:

PROPOSITION 28. *Let $\mathcal{S}$ be an order-normalizing transformation system wrt. $\mathcal{N}$ and let $\mathcal{I}$ be a completed inference system wrt. $\mathcal{S}$. Whenever there exists a ground derivation $D = (L \Longrightarrow^*_{\mathcal{I}^g,M} L')$ then there also exists a ground derivation $D' = (L \Longrightarrow^*_{\mathcal{I}^g,M} L')$ with $D' \in \mathcal{N}$ and $D' \preceq D$, where $\succ$ is the derivation ordering of $\mathcal{S}$.*

Proof. By well-founded induction on the derivation ordering $\succ$: either $D \in \mathcal{N}$ and we are done by taking $D' = D$, or else by definition of order-normalizing transformation systems there exists a derivation $D'' = (L \Longrightarrow^*_{(\mathcal{I}')^g,M} L')$ with $D'' \prec D$, where $\mathcal{I}' = \mathcal{I}$ or $\mathcal{I} \vdash_{\mathcal{S}} \mathcal{I}'$ by application of some mandatory transformation rule. If $\mathcal{I}' = \mathcal{I}$ we can immediately apply the induction hypothesis to D'' to obtain the desired derivation D'. Otherwise, $\mathcal{I}' = \mathcal{I} \cup \{P \to C\}$. Since $\mathcal{I}$ is completed either (a) a variant of $P \to C$ is contained in $\mathcal{I}$, or (b) $P \to C$ is $\succ$-redundant in $\mathcal{I}$. In case (a) we can replace in D'' every use of $P \to C$ by its variant from $\mathcal{I}$ and obtain a $\mathcal{I}$-derivation alone (to which the induction hypothesis can be applied). Now for case (b): either $P \to C$ is not used and thus D'' is a $\mathcal{I}$-derivation alone (to which the induction hypothesis can be applied), or else, by definition of redundancy (Def. 10) there exists a $\mathcal{I}^g$-derivation $D''' \prec D''$ of the same kind as D''. By applying the induction hypothesis to D''' this concluding case is done.

Chaining ground completeness (Lemma 4), the completion property of limits (Theorem 25) with Proposition 28 we can obtain normal refutations of $M \cup punit(\mathcal{I})$, where M is some $\mathcal{T}$-unsatisfiable literal set. Thus, a refutation might still use unit inference rules being turned via $punit(\mathcal{I})$ into input literals. However, every use of a literal from $punit(\mathcal{I})$ represents a computation among the inference rules and should be avoided. In order to admit normal refutations with input literals M alone, we will compile the literals $punit(\mathcal{I})$ into the inference system. The situation is much like normalizing above, but now "normal" means "free of usages of elements of $punit(\mathcal{I})$". In order to obtain termination when eliminating these cases we require the respective mandatory transformation rules to work strictly decreasing wrt. $\succ$. This is captured in the next definition.

DEFINITION 29. **(Punit-Normalizing Transformation System)** The multiset of *used input literals of a derivation D* is defined as

$$used(D) := \{\!\{L \mid L \text{ is the top literal of } D|_p, \text{ where } p \in P(D)\}\!\}$$

The function $used$ is extended homomorphically to sequences and multisets of derivations as expected.

Let $\mathcal{S}$ be a transformation system with derivation ordering $\succ$ and let $\mathcal{N}$ be a set of normal derivations. Then $\mathcal{S}$ is called *punit-normalizing wrt.* $\mathcal{N}$ iff whenever there exists a non-trivial ground derivation

$$D = (L \Longrightarrow^*_{\mathcal{I}^g, M \cup punit(\mathcal{I}^g)} L')$$

with $D \in \mathcal{N}$ such that $used(D) \cap punit(\mathcal{I}^g) \neq \emptyset$ then there exists a derivation

$$D' = (K \Longrightarrow^*_{(\mathcal{I}')^g, M \cup punit((\mathcal{I}')^g)} L')$$

with $D' \prec D$ where $K \in M \cup punit((\mathcal{I}')^g) \cup \{L\}$ and $\mathcal{I}' = \mathcal{I}$ or $\mathcal{I} \vdash_{\mathcal{S}} \mathcal{I}'$ by application of some mandatory transformation rule.

"punit-normalization" means that whenever an inference rule $L \rightarrow false$ is used as as an input literal $\overline{L}$ then a strictly smaller derivation exists. That derivation, however, need not start with the same top literal L, but may as well start with a top literal from $M \cup punit((\mathcal{I}')^g)$. Note that according to this definition it suffices for a punit-normalizing transformation system to work on normal-form derivations only.

As an example take again "linearizing completion", which is punit-normalizing wrt. linear ground derivations. This is essentially due to the transformation rules **Unit1** and **Unit2**. See also example 15.

PROPOSITION 30. *The transformation system Lin is punit-normalizing wrt. LinG.*

Completed inference systems in conjunction with punit-normalizing transformation systems yield derivations that are free of applications of inference rules $L \rightarrow false$ as input literals $\overline{L}$. This result is similar to Proposition 28 for order-normal derivations:

PROPOSITION 31. *Let $\mathcal{S}$ be a punit-normalizing transformation system wrt. $\mathcal{N}$, and let $\mathcal{I}$ be a completed inference system wrt. $\mathcal{S}$. Whenever there exists a ground derivation $D = (L \Longrightarrow^*_{\mathcal{I}^g, M \cup punit(\mathcal{I}^g)} L')$ with $D \in \mathcal{N}$ then there also exists a ground derivation $D' = (K \Longrightarrow^*_{\mathcal{I}^g, M} L')$ with $D' \preceq D$, $D' \in \mathcal{N}$ and some $K \in M$.*

Since we are interested in both properties — punit-normalization and order-normalization — we define:

DEFINITION 32. A transformation system $\mathcal{S}$ is called *normalizing wrt. $\mathcal{N}$* iff $\mathcal{S}$ is both order-normalizing wrt. $\mathcal{N}$ and punit-normalizing wrt. $\mathcal{N}$.

Combining Propositions 27 and 30 we then obtain:

THEOREM 33. *The Transformation system Lin is normalizing wrt. the set LinG of linear derivations.*

8. Completeness

The goal of this section is to assemble the material of the previous sections into several completeness results. We will prove two versions of general ground completeness results. "General" here means that no special derivation ordering is supposed. These results then will be instantiated for the case of linearizing completion and lifted to the first-order level in Section 8.3.

8.1. Ground Completeness

In purely *equational* logic and Knuth-Bendix completion, Birkhoff's theorem links model theory and proof theory: two ground terms are equal in an equational theory $\mathcal{T}$ — i.e. a set of equations — iff they can be made identical by replacement operations using the equations from $\mathcal{T}$. Since we deal with more general theories we proved in Section 3 a corresponding result (ground completeness, Lemma 4). In order to apply it we find it helpful to introduce the following notion:

DEFINITION 34. **(Relative Completeness)** An inference system $\mathcal{I}$ is called *relatively complete wrt. an inference system $\mathcal{J}$* iff whenever

$$L \Longrightarrow^*_{\mathcal{J}^g, M \cup punit(\mathcal{J}^g)} L'$$

then also

$$L \Longrightarrow^*_{\mathcal{I}^g, M \cup punit(\mathcal{I}^g)} L'$$

Now we can turn towards completeness. We start with a general result:

THEOREM 35. **(General Ground Completeness Theorem)** *Let $\mathcal{T}$ be a theory and let $\mathcal{S}$ be a normalizing transformation system wrt. some set $\mathcal{N}$ of normal derivations. Suppose an inference system $\mathcal{I}$ is completed wrt. $\mathcal{S}$, and also suppose that $\mathcal{I}$ is relatively complete wrt. $\mathcal{I}_0(\mathcal{T})$. Then for every $\mathcal{T}$-unsatisfiable ground literal set M there exists a $\mathcal{I}^g$-refutation $D \in \mathcal{N}$ of M with some top literal from M.*

Proof. M is $\mathcal{T}$-unsatisfiable iff M is $\mathcal{T}^g$-unsatisfiable. By ground completeness (Lemma 4) $L \Longrightarrow^*_{\mathcal{I}_0(\mathcal{T}^g), M \cup punit(\mathcal{I}_0(\mathcal{T}^g))} false$ for some $L \in M \cup punit(\mathcal{I}_0(\mathcal{T}^g))$. With $\mathcal{I}_0(\mathcal{T}^g) \subseteq \mathcal{I}_0(\mathcal{T})^g$ it follows $L \Longrightarrow^*_{\mathcal{I}_0(\mathcal{T})^g, M \cup punit(\mathcal{I}_0(\mathcal{T})^g)} false$. Since $\mathcal{I}$ is given as relatively complete wrt. $\mathcal{I}_0(\mathcal{T})$ we find by definition
$L \Longrightarrow^*_{\mathcal{I}^g, M \cup punit(\mathcal{I}^g)} false$. With $\mathcal{I}$ being given as normalizing wrt. $\mathcal{N}$ we can first find (by Proposition 28) an order-normal refutation $D' = (L \Longrightarrow^*_{\mathcal{I}^g, M\gamma \cup punit(\mathcal{I}^g)} false) \in \mathcal{N}$ and then we can find (by Proposition 31) a punit-normal refutation $D = (K \Longrightarrow^*_{\mathcal{I}^g, M} false \in \mathcal{N})$ for some $K \in M$.

This theorem requires the existence of a completed and relatively complete inference system wrt. $\mathcal{I}_0(\mathcal{T})$. Such a system can be obtained in a constructive way as the limit of a fair deduction:

THEOREM 36. *The limit $\mathcal{I}_\infty$ of a $\mathcal{S}$-deduction $\mathcal{I}_0(T) \vdash_\mathcal{S} \mathcal{I}_1 \vdash_\mathcal{S} \mathcal{I}_2 \ldots$ is relatively complete wrt. $\mathcal{I}_0(\mathcal{T})$, where $\mathcal{T}$ is a theory.*

Proof. Use Lemma 22, setting there $\mathcal{I}_0 = \mathcal{I}_0(\mathcal{T})$ and $k = 0$.

Thus we can instantiate the general ground completeness theorem:

COROLLARY 37. *Let $\mathcal{T}$, $\mathcal{S}$ and $\mathcal{N}$ as in Theorem 35, and let $\mathcal{I}_\infty$ be the limit of a fair $\mathcal{S}$-deduction $\mathcal{I}_0(\mathcal{T}) \vdash_\mathcal{S} \mathcal{I}_1 \vdash_\mathcal{S} \ldots$. Then for every $\mathcal{T}$-unsatisfiable ground literal set M there exists a $\mathcal{I}^g_\infty$-refutation $D \in \mathcal{N}$ of M with some top literal from M.*

Proof. By Theorem 36 $\mathcal{I}_\infty$ is relatively complete wrt. $\mathcal{I}_0(\mathcal{T})$, and by Theorem 25 $\mathcal{I}_\infty$ is completed wrt. $\mathcal{S}$. Hence the corollary follows from the theorem.

Next we are going to instantiate this corollary towards "linearizing completion". Before we do so observe that the corollary (as well as the theorem) states completeness for *some* top literal chosen from the input literal set. However, as motivated in the introduction, the intended application of completed inference systems as background reasoners within theory reasoning calculi demands that we insist on a completeness result with respect to *every* literal as top literal (see (Baumgartner, 1994) for a detailed treatment). This "independence of the goal" property corresponds to e.g. linear resolution for Horn theories where it suffices to start derivations with a negative clause. In practice this means that the top literal need not be guessed in a don't-know nondeterministic way, but can be chosen a priori.

Fortunately, the transformation system Lin is powerful enough to generate inference rules that allow rearrangement of a given linear derivation such that any literal used

inside a derivation can be switched to the top position. This is expressed in the following lemma:

LEMMA 38. **(Top literal lemma)** *Let $\mathcal{I}$ be a completed inference system wrt. the transformation system Lin. Suppose there exists a linear ground derivation $D = (L_1 \Longrightarrow^*_{\mathcal{I}^g,M} L_n)$ with $L_1 \in M$. Let $T \in M$ such that $T \in used(M)$. Then there exists a linear ground derivation $D' = (T \Longrightarrow^*_{\mathcal{I}^g,M} L_n)$.*

For instance, the inference system $\mathcal{I}_\infty(\mathcal{T})$ in the introduction is completed wrt. $LinG$. Now consider again the refutation R_1 of $\mathcal{L}$ (also given in the introduction)[12]. Since $a < c \in used(\mathcal{L})$ there exists a refutation with top literal $a < c$, which is R_2.

Note that the top literal lemma is not formulated within the general uninstantiated completion theory, since in general it may be not useful to demand the completeness for arbitrary chosen top literals for arbitrary transformation systems.

Now with the top literal lemma we can obtain the desired completeness result for linearizing completion:

THEOREM 39. **(Ground Completeness of "Linearizing Completion")** *Let $\mathcal{I}$ be an inference system completed wrt. the transformation system Lin. Let $\mathcal{T}$ be a theory and suppose $\mathcal{I}$ is relatively complete wrt. $\mathcal{I}_0(\mathcal{T})$. Then for every minimal $\mathcal{T}$-unsatisfiable ground literal set M and every literal $L \in M$ there exists a linear refutation*

$$D = (L \Longrightarrow^*_{\mathcal{I}^g,M} \mathit{false})$$

Note that a relatively complete inference system as assumed in the theorem can be obtained by application of Theorem 36.

Proof. By the general ground completeness theorem (Theorem 35) there exists a linear $\mathcal{I}^g$-refutation $D \in LinG$ of M. Since M is given as minimal $\mathcal{T}$-unsatisfiable, the intended top literal L must be used at least once in the refutation (because otherwise the refutation of $M \setminus \{L\}$ implies by soundness a contradiction to the minimality assumption about M). Next apply the top literal lemma (Lemma 38) to D and obtain a linear refutation with top literal L.

By the theorems and corollaries of this section the main results of our completion technique for the ground case are established.

8.2. Theory Complementarity and Theory Refuters

First order completeness of refutational proof calculi usually is formulated wrt. (theory-) unsatisfiable input formulas. This is fine if one is interested only in simple "yes/no"-answers. However, as mentioned in the introduction, our inference systems shall be used as background reasoners within theory reasoning calculi (such as theory model elimination or theory resolution). For this task it is necessary to compute an *answer substitution* by

[12] Although not ground, it will serve as an example equally well.

the background reasoner to be passed back to the foreground reasoner. Thus we cannot be content with refutational completeness alone. On the other hand, we obtain traditional refutational completeness as a corollary of the completeness wrt. answer substitutions.

In non-theory calculi, the refutations are usually computed for efficiency reasons at a most general level. This is also a desirable goal within theory reasoning. But then, however, unifiers need no longer be unique, and thus have to be replaced by a more general concept. In the presence of purely *equational* theories the concept of "complete set of unifiers" is well-known. Since we deal with arbitrary Horn theories and not just equational theories we still have to be a bit more general. This concept, which is formulated in a dual way, is called *theory refuting substitutions*. Below we will show that our inference systems can be used to compute most general theory refuting substitutions.

DEFINITION 40. **(Theory complementary, Theory refuting substitution)** A literal set $M = \{L_1, \ldots, L_n\}$ is called *$\mathcal{T}$-complementary*[13] iff the existentially quantified formula $\exists(L_1 \wedge \ldots \wedge L_n)$ is $\mathcal{T}$-unsatisfiable. M is called *minimal $\mathcal{T}$-complementary* iff M is $\mathcal{T}$-complementary and all proper subsets are not $\mathcal{T}$-complementary.

A substitution σ is a (minimal) $\mathcal{T}$-refuter for a literal set M iff $M\sigma$ is (minimal) $\mathcal{T}$-complementary. A set of substitutions is a *complete and most general set of $\mathcal{T}$-refuting substitutions for M* (or short: $CSR_{\mathcal{T}}(M)$) iff

1. for all $\sigma \in CSR_{\mathcal{T}}(M)$: $M\sigma$ is $\mathcal{T}$-complementary *(Correctness)*
2. for all substitutions θ such that $M\theta$ is $\mathcal{T}$-complementary:
   there exists a $\sigma \in CSR_{\mathcal{T}}(M)$ and a substitution σ' such that $\theta = \sigma\sigma'|dom(\theta)$
   (Completeness)

The members of $CSR_{\mathcal{T}}(M)$ are also called *most general $\mathcal{T}-$refuters for M*.

Our notion of theory refuter generalizes the notion of *rigid E-unifier* (Gallier *et al.*, 1990) to more general theories than equality (see (Baumgartner, 1992b)). A dual notion, "unifier with respect to $\mathcal{T}$-complementary literal sets", has been studied within an affirmative setting in (Petermann, 1991).

EXAMPLE 41. Consider the theory $\mathcal{E}$ of equality. The set

$$M = \{P(x),\ y = f(y),\ \neg P(f(z))\}$$

clearly is $\mathcal{E}$-unsatisfiable when read as a set of unit clauses. However it is not $\mathcal{E}$-complementary, since e.g. the ground instance $P(a) \wedge (a = f(a)) \wedge \neg P(f(b))$ of

$$\exists x, y, z : (P(x) \wedge (y = f(y)) \wedge \neg P(f(z)))$$

is not $\mathcal{E}$-unsatisfiable. But the substitution $\sigma = \{x \leftarrow z, y \leftarrow z\}$ is a $\mathcal{E}$-refuter since the formula $\exists z : (P(z) \wedge z = f(z) \wedge \neg P(f(z)))$ obtained from $M\sigma$ is $\mathcal{E}$-unsatisfiable (since there does not exist a ground instance that is $\mathcal{E}$-satisfiable).

[13] this definition is intended as a generalization of standard "syntactic complementarity" which means that two literals are syntactically complementary iff one of them is the negation of the other.

8.3. FIRST ORDER COMPLETENESS

First of all, the definition of "derivation" has to be lifted to the first order case:

DEFINITION 42. A *first-order inference* is defined in the same way as an inference (cf. Section 3), except that the substitution δ is replaced by a (Multiset-)MGU σ for P' and P. More precisely, we say that a literal C' *is first-order inferred from a literal multiset* P' *by an inference rule* $P \rightarrow_w C$ *and substitution* σ, written as a *first-order inference*

$$P' \Longrightarrow_{P \rightarrow_w C, \sigma} C'$$

iff $P'\sigma = P\sigma$ and $C\sigma = C'$.

A *linear first-order derivation of* M is defined in the same way as a linear derivation in Section 3, except that the inferences $\{\!|L_i, L_i^1, \ldots, L_i^{m_i}|\!\} \Longrightarrow_{P_i \rightarrow C_i, \delta_i} L_{i+1}$ are replaced by first-order inferences $\{\!|L_i, L_i^1, \ldots, L_i^{m_i}|\!\} \Longrightarrow_{P_i \rightarrow C_i, \sigma_i} L_{i+1}$ where $L_i^1, \ldots, L_i^{m_i} \in M\sigma_1 \cdots \sigma_{i-1}$ (thus in every step the so far computed substitution $\sigma_1 \cdots \sigma_{i-1}$ is applied to the input set M). Furthermore the used inference rule $P_i \rightarrow C_i$ has to be a new variant of some rule from $\mathcal{I}$. The *answer substitution* σ of a linear first-order derivation is defined as $\sigma := \sigma_1\sigma_2 \cdots \sigma_{n-1}|_{var(M)}$ if $n > 1$, or else $\sigma := \varepsilon$.

A *(linear first-order)* $\mathcal{I}$*-refutation* is defined as a (linear first-order) $\mathcal{I}$-derivation of $false$.

We write $L_1 \Longrightarrow^{\dagger}_{\mathcal{I},M,\sigma} L_n$ to indicate that a linear first-order $\mathcal{I}$-derivation with answer substitution σ of L_n from input literals M exists. Furthermore the notation $D = (L_1 \Longrightarrow^{\dagger}_{\mathcal{I},M,\sigma} L_n)$ means that D is such a derivation.

First order completeness is shown in a standard way which employs a lifting lemma for derivations. However, a dedicated lifting lemma has to be proved for the derivation ordering in use. Hence, first-order completeness will be formulated wrt. a derivation ordering. Here we will show the case for *linear refutations*.

LEMMA 43. **(Lifting lemma for linear refutations)** *Let* L_1 *be a literal,* M *be a literal set and* γ *be a ground substitution for* L_1 *and* M*. If there exists a non-trivial linear refutation* $L_1\gamma \Longrightarrow^{*}_{\mathcal{I}^g,M\gamma}$ *false then there exists a linear first-order refutation* $L_1 \Longrightarrow^{\dagger}_{\mathcal{I},M,\sigma}$ *false such that* $\sigma \leq \gamma \quad [var(M)]$.

Finally we obtain the completeness theorem for the case of linearizing completion; it is the main result of this paper, and its proof employs most of the material presented here.

THEOREM 44. *Let* $\mathcal{I}$ *be an inference system completed wrt. the transformation system Lin. Let* $\mathcal{T}$ *be a theory and suppose* $\mathcal{I}$ *is relatively complete wrt.* $\mathcal{I}_0(\mathcal{T})$*. Let* M *be a minimal* $\mathcal{T}$*-complementary literal set,* $L \in M$*, and suppose* γ *is a* $\mathcal{T}$*-refuter for* M*. Then there exists a linear refutation*

$$D^{\dagger} = (L \Longrightarrow^{\dagger}_{\mathcal{I},M,\sigma} \mathit{false})$$

with computed answer substitution $\sigma \leq \gamma \quad [var(M)]$.

Proof. Since γ is a minimal $\mathcal{T}$-refuter, $M\gamma$ is minimal $\mathcal{T}$-complementary. Hence also the ground instance $L\gamma$ of $L \in M$ is contained in $M\gamma$.

By definition, $M\gamma$ is $\mathcal{T}$-complementary iff $\exists M\gamma$ is $\mathcal{T}$-unsatisfiable (where M is read as a conjunction). Clearly, all existentially quantified variables can now be skolemized away, which preserves $\mathcal{T}$-unsatisfiablity. Let γ' be such a substitution which replaces all variables in $M\gamma$ by new constants. Hence $M\gamma\gamma'$ is ground and is $\mathcal{T}$-unsatisfiable. By the ground completeness of "linearizing completion" (Theorem 39) there exists a linear $\mathcal{I}^g$-refutation of $M\gamma\gamma'$ with top literal $L\gamma\gamma'$. Finally apply the lifting lemma (Lemma 43) to obtain the linear first-order refutation $D^\dagger$ of L.

9. Practical Experiments

We have implemented linearizing completion and combined it with our PTTP-based theory model elimination prover, called PROTEIN (PROver with a Theory Extension Interface, (Baumgartner and Furbach, 1994b)). Besides the more straightforward extension of model elimination towards theory reasoning, PROTEIN also features a variant which relies on "case-analysis" reasoning similar to Loveland's Near-Horn Prolog or Plaisted's modified problem reduction format. This variant is called *restart model elimination* ((Baumgartner and Furbach, 1994a)). Both PROTEIN and the linearizing completion tool are implemented in ECLiPSe, ECRC's Prolog dialect.

We ran several examples known from the literature with PROTEIN and another high-performance model elimination prover, SETHEO V. 3.0 (Letz *et al.*, 1992). Table 12 contains the runtime results (in seconds), obtained on a SPARC 10/40. The first four columns refer to different versions of PROTEIN. Column 5 contains data for SETHEO.

PROTEIN was run in default mode, except where indicated in Table 12. In default mode it includes the regularity restriction and the ground-reduction refinement. SETHEO was also run in its default mode, which then makes use of the following refinements and constraints: subgoal reordering, purity, anti-lemmas, regularity, tautology and subsumption. Concerning the search strategy we used iterative deepening with the costs of extension steps uniformly set to 1. The same costs are used for case analysis steps.

For the theory variants of PROTEIN, the background calculus was obtained automatically by the linearizing completion tool in a preprocessing phase. In this setting, PROTEIN implements the partial theory model elimination calculus as described in the introduction.

For the theories we have selected appropriate Horn-subsets of the input clauses. These were in most cases "natural" theories (such as equality and orderings). We observed that it is important not to select a theory which would result in a (partially) completed inference system of more than about 150 inference rules. In such cases the local search space is too broad in order to be explored to significant depth.

Example	ME	Restart-ME	TME	Restart-TME	SETHEO
Non-obvious MSC/MSC006-1	0.3	2.7	1.6	1.1 0.15 [1]	0.5
Graph	10.8	∞	0.2	7.0 0.8 [2]	
$x \neq 0 \rightarrow x^2 > 0$	2.4	0.7	2.2	0.6	0.8
Pelletier 48 SYN/SYN071-1	5.9	1.2 0.6 [2]	0.4	0.9 0.1 [1,2]	0.2
Pelletier 49 SYN/SYN072-1	∞	297	1.6	1.5	∞
Pelletier 55 PUZ/PUZ001-2	392	∞	21	254	3.5
Lion&Unicorn PUZ/PUZ005-1	588	∞	21	∞	47
Wos 4 GRP/GRP008-1	22	20	0.3	26	13
Wos 10 GRP/GRP001-1	∞	-	14	-	850
Wos 11 GRP/GRP013-1	9.6	-	1.1	-	0.7
Wos 15 GRP/GRP035-3	384	-	47	-	478
Wos 16 GRP/GRP036-3	302	-	0.02	-	13
Wos 17 GRP/GRP037-3	∞	-	0.1	-	23

Entries: Numbers: runtimes (seconds) – ∞ no proof within reasonable time bound – "-" Not applicable –
Remarks: 1 – With selection function, 2 – With (back) factoring,

Fig. 12. Runtime Results for various provers: *ME* – plain model elimination version of PROTEIN; *Restart-ME* – case-analysis style reasoning; *TME* and *Restart-TME* – respective versions with theory reasoning extensions.

The weight of every inference rule in the initial system is set to 0. This value was chosen heuristically. In general, lower values for weights imply that it is less likely that the rule will be detected as redundant, while it is more likely that the rule is used in the redundancy proof of a different rule. Higher values imply the opposite behavior. The weights of the generated inference rules were determined heuristically: when newly generated, an inference rule becomes the highest admissible weight such that redundancy proofs using this rule still succeed. It should be noted that these heuristics are implemented, and no user assignment of weights is necessary.

The example referred to as *Non-obvious* is taken from the October 1986 Newsletter of the *Association of Automated Reasoning*[14]. The selected theory here consists of a

[14] Entries such as MSC/MSC006-1 refer to the respective TPTP-names (Sutcliffe *et al.*, 1994). All examples were drawn from that problem library (Version 1.1.0) without modification — only the theory part had to be selected by hand.

transitive and symmetric relation p and a transitive relation q. It can be completed to a finite system in a fully automated way. In the *Graph* example a graph with a transitive and symmetric reachability relation is traversed. The example referred to by $x \neq 0 \rightarrow x^2 > 0$ is to prove this theorem (x is universally quantified) from calculus. Case analysis is carried out according to the axiom $X > 0 \quad \vee \quad X = 0 \quad \vee \quad -X > 0$. The theory part of the Pelletier examples consists of an equivalence relation and the completed system is finite. The Wos examples are from group theory. Notably, it suffices to use the *same* theory to prove all examples. Here we took equality (except function substitution axioms) and the associativity of the group operation.

The runtime of the linearizing completion tool was either sufficiently small and need not be mentioned (in case we have theories which are special to one single example, as for Non-obvious), or else the selected theory works for a whole class of examples (as for the Wos examples) and its completion can be viewed as being done "beforehand". In case linearizing completion would yield an infinite inference system for background reasoning – in the Wos examples from group theory – a finite approximation was used. All examples could be proved with the same finite approximation.

10. Conclusions

In this paper we have developed a new completion technique for Horn theories that allows for the combination of the linear and unit-resulting restrictions. The central operation is to add new inference rules that detour violations of the linearity restrictions in unit-resulting refutations. A redundancy criterion allows deletion of many of the thus added inference rules, which makes the resulting inference systems quite compact.

The completed inference systems can be used as efficient background reasoners within theory reasoning calculi. The method is fully implemented and runs in cooperation with our theory model elimination theorem prover PROTEIN[15]. On numerous examples drastic speedups were obtained. Notably, the method works fine not only for single selected examples, but also for whole classes of examples. We demonstrated this by completing a subset of group theory for the Wos examples. Surely, other classes will have to be identified in the future.

Some more notes on further work: it is possible to allow in derivation steps accessing of previously derived literals as side literals. This corresponds to "ancestor resolution" steps in ordinary linear resolution. As a gain of this modification more inference rules will become redundant.

Currently, the method is limited to Horn theories. It might be worthwile to design an extension towards general, non-Horn theories. From the technical point of view, the problem is that unit-resulting resolution is not complete for this case. Hence we get a gap in the completeness proof. It is conceivable that splitting into Horn theories helps here.

[15] The whole system is available in the World Wibe Web, using the URL `http://www.uni-koblenz.de/~peter/protein.html`.

The *linked inference principle* (Veroff and Wos, 1992), is a resolution inference rule in the spirit of hyperresolution. In its Unit-resulting variant, an inference steps consists of the construction of a whole resolution tree with a one-literal clause as conclusion. This search at run-time could possibly be supplanted by our completion-based theory compilation. However, this is not yet worked out.

We think the method is general enough to be applicable to other ordering criteria; for example one might think of term-ordering restrictions as applied in the term rewriting paradigm, or the combination of term-ordering restrictions and linearity restrictions. Of course, different restrictions require different transformation systems, but many of the concepts and claims not related to a specific restrictions can be kept. This will be done in the near future.

The availability of unification algorithms for dedicated theories motivates us to extend the method towards "completion modulo a built-in theory" E. By this, the combined theory consisting of the Horn theory and E would be subject to theory reasoning. In order to do so one has to use E-unification instead of syntactic unification during the completion phase *and* in inferences using the completed system.

Finally, we are currently implementing a library of completed theories. It consists of various kinds of orderings, equality and group theory. In order to facilitate its use, we are implementing a program which scans an input file for respective axioms and replaces them by the theory inference rules from the library.

Acknowledgments: I thank J. Dix, U. Furbach, O. Menkens and F. Stolzenburg for reading earlier drafts of this paper. I am very indebted to three anonymous referees who read the paper with great care and suggested numerous improvements.

References

Owen L. Astrachan and Mark E. Stickel. Caching and Lemmaizing in Model Elimination Theorem Provers. In D. Kapur, editor, *Proceedings of the 11th International Conference on Automated Deduction (CADE-11)*, pages 224–238. Springer-Verlag, June 1992. LNAI 607.

L. Bachmair and H. Ganzinger. On Restrictions of Ordered Paramodulation with Simplification. In M.E. Stickel, editor, *Proc CADE 10, LNCS 449*, pages 427–441. Springer, 1990.

L. Bachmair and H. Ganzinger. Rewrite Techniques for Transitive Relations. Research Report MPI-I-93-249, Max-Planck-Institut für Informatik, 1993.

Leo Bachmair, Nachum Dershowitz, and Jieh Hsiang. Orderings for equational proofs. *IEEE*, pages 346–357, 1986.

L. Bachmair, N. Dershowitz, and D. Plaisted. Completion without failure. In H. Ait-Kaci and M. Nivat, editors, *Resolution of Equations in Algebraic Structures 2: Rewrite Techniques*, pages 1–30. Academic Press, 1989.

L. Bachmair, H. Ganzinger, C. Lynch, and W. Snyder. Basic Paramodulation and Superposition. In D. Kapur, editor, *Proc. 11th CADE*, pages 462–476. Springer, 1992.

Leo Bachmair. *Proof Methods for Equational Theories.* PhD thesis, University of Illinois at Urbana, U.S.A., 1987.

L. Bachmair. *Canonical Equational Proofs.* Progress in Theoretical Computer Science. Birkhäuser, 1991.

P. Baumgartner and U. Furbach. Consolution as a Framework for Comparing Calculi. *Journal of Symbolic Computation*, 16(5), 1993. Academic Press.

P. Baumgartner and U. Furbach. Model Elimination without Contrapositives. In A. Bundy, editor, *'Automated Deduction – CADE-12'*, volume 814 of *LNAI*, pages 87–101. Springer, 1994.

P. Baumgartner and U. Furbach. PROTEIN: A *PRO*ver with a *T*heory *E*xtension *I*nterface. In A. Bundy, editor, *Automated Deduction – CADE-12*, volume 814 of *LNAI*, pages 769–773. Springer, 1994.

P. Baumgartner, U. Furbach, and U. Petermann. A Unified Approach to Theory Reasoning. Research Report 15/92, University of Koblenz, 1992.

P. Baumgartner. A Model Elimination Calculus with Built-in Theories. In H.-J. Ohlbach, editor, *Proceedings of the 16-th German AI-Conference (GWAI-92)*, pages 30–42. Springer, 1992. LNAI 671.

P. Baumgartner. An Ordered Theory Resolution Calculus. In A. Voronkov, editor, *Logic Programming and Automated Reasoning (Proceedings)*, pages 119–130, St. Petersburg, Russia, July 1992. Springer. LNAI 624.

P. Baumgartner. Refinements of Theory Model Elimination and a Variant without Contrapositives. In A.G. Cohn, editor, *11th European Conference on Artificial Intelligence, ECAI 94*. Wiley, 1994. (Long version in: Research Report 8/93, University of Koblenz, Institute for Computer Science, Koblenz, Germany).

Hubert Bertling. Knuth-Bendix Completion of Horn Clause Programs for Restricted Linear Resolution and Paramodulation. In S. Kaplan and M. Okada, editors, *Proceedings of the 2nd International Workshop on Conditional and Typed Rewriting Systems*, pages 181–193. Springer-Verlag, June 1990. LNCS 516.

W. W. Bledsoe. Challenge Problems in Elementary Calculus. *Journal of Automated Reasoning*, 6:341–359, 1990.

T. Bollinger. A Model Elimination Calculus for Generalized Clauses. In *IJCAI*, 1991.

Francois Bronsard and Uday S. Reddy. Reduction Techniques for First-Order Reasoning. In M. Rusinowitch and J.L. Rémy, editors, *Proceedings of the Third International Workshop on Conditional Term Rewriting Systems*, pages 242–256. Springer-Verlag, July 1992. LNCS 656.

H.-J. Bürckert. *A Resolution Principle for Clauses with Restricted Quantifiers*, volume 568 of *LNAI*. Springer, 1991.

W. Büttner. Unification in the Datastructure Multisets. *Journal of Automated Reasoning*, 2:75–88, 1986.

C. Chang and R. Lee. *Symbolic Logic and Mechanical Theorem Proving*. Academic Press, 1973.

C. Chang and R. Lee. *Symbolic Logic and Mechanical Theorem Proving*. Academic Press, 1973.

N. Dershowitz and Z. Manna. Proving Termination with multiset orderings. *Comm. ACM*, 22:465–476, 1979.

N. Dershowitz. Orderings for Term-Rewriting Systems. *Theoretical Computer Science*, 17:279–301, 1982.

N. Dershowitz. Computing with Rewrite Systems. *Information and Control*, 65:122 – 157, 1985.

Nachum Dershowitz. Termination of Rewriting. *Journal of Symbolic Computation*, 3(1&2):69–116, February/April 1987.

Nachum Dershowitz. A Maximal-Literal Unit Strategy for Horn Clauses. In S. Kaplan and M. Okada, editors, *Proceedings of the 2nd International Workshop on Conditional and Typed Rewriting Systems*, pages 14–25. Springer-Verlag, June 1990. LNCS 516.

N. Dershowitz. Ordering-Based Strategies for Horn Clauses. In *Proc. IJCAI*, 1991.

Nachum Dershowitz. Canonical Sets of Horn Clauses. In J. Leach Albert, B. Monien, and M. Rodríguez Artalejo, editors, *Proceedings of the 18th International Colloquium on Automata, Languages and Programming*, pages 267–278, July 1991. LNCS 510.

Vincent J. Digricoli and Malcolm C. Harrison. Equality-Based binary Resolution. *Journal of the Association for Computing Machinery*, 1986.

J. Dixon. Z-Resolution: Theorem-Proving with Compiled Axioms. *Journal of the ACM*, 20(1):127–147, 1973.

Ulrich Furbach, Steffen Hölldobler, and Joachim Schreiber. Horn equational theories and paramodulation. *Journal of Automated Reasoning*, 3:309–337, 1989.

J. H. Gallier, S. Raatz, and W. Snyder. Theorem proving using rigid e-unification: Equational matings. In *Logics in Computer Science '87, Ithaca, New York*, 1987.

J. Gallier, P. Narendran, D. Plaisted, and W. Snyder. Rigid E-unification: NP-Completeness and Applications to Equational Matings. *Information and Computation*, pages 129–195, 1990.

Harald Ganzinger. A Completion Procedure for Conditional Equations. *Journal of Symbolic Computation*, 11:51–81, 1991.

L. Hines. Str+ve$\subseteq$: The Str+ve-based Subset Prover. In M. Stickel, editor, *Proceedings of the 10th International Conference on Automated Deduction (CADE-10)*, pages 193–206. Springer-Verlag, 1990. LNAI 449.

L. Hines. The Central Variable Strategy of Str+ve. In D. Kapur, editor, *Proceedings of the 11th International Conference on Automated Deduction (CADE-11)*, pages 35–49. Springer-Verlag, June 1992. LNAI 607.

Steffen Hölldobler. *Foundations of Equational Logic Programming*, volume 353 of *Lecture Notes in Artificial Intelligence. Subseries of Lecture Notes in Computer Science*. Springer, 1989.

J. Hsiang and N. Dershowitz. Rewrite Methods for Clausal and Non-Clausal Theorem Proving. In *Proc. ICALP'83*, 1983.

J. Hsiang and M. Rusinowitch. On word problems in equational theories. In *Proc. ICALP'87*, pages 54–71. Springer, LNCS 267, 1987.

Gérard Huet and Derek C. Oppen. Equations and Rewrite Rules. A Survey. In R. Book, editor, *Formal Languages: Perspectives and Open Problems*, pages 349–405. Academic Press, 1980.

J.M. Hullot. Canonical forms and unification. In *Proc. Conf. Automated Deduction*, pages 318–334, 1980.

Stéphane Kaplan. Simplifying Conditional Term Rewriting Systems: Unification, Termination and Confluence. *J. of Symbolic Computation*, 4(3):295–334, 1987.

Donald E. Knuth and B. Bendix, Peter. Simple world problems in universal algebras, 1970.

R. Letz, J. Schumann, S. Bayerl, and W. Bibel. SETHEO: A High-Performance Theorem Prover. *Journal of Automated Reasoning*, 8(2), 1992.

R. Letz, K. Mayr, and C. Goller. Controlled Integrations of the Cut Rule into Connection Tableau Calculi. Journal of Automated Reasoning (to appear 1994), 1993.

J. Lloyd. *Foundations of Logic Programming*. Springer, 1984.

D. Loveland. Mechanical Theorem Proving by Model Elimination. *JACM*, 15(2), 1968.

D. Loveland. A Linear Format for Resolution. In *Symposium on Automatic Demonstration*, number 125 in Lecture Notes in Mathematics, pages 147–162, 1970.

D. Loveland. *Automated Theorem Proving - A Logical Basis*. North Holland, 1978.

Z. Manna and R. Waldinger. Special Relations in Automated Deduction. *Journal of the ACM*, 33(1):1–59, 1986.

Z. Manna, M. Stickel, and R. Monotonicity Properties in Automated Deduction. In V. Lifschitz, editor, *Artificial Intelligence and Mathematical Theory of Computation: Essays in Honor of John McCarthy*, pages 261–280. Academic Press, 1991.

J. McCharen, R. Overbeek, and L. Wos. Complexity and related enhancements for automated theorem-proving programs. *Computers and Mathematics with Applications*, 2:1–16, 1976.

J. B. Morris. E-Resolution: An Extension of Resolution to include the Equality Relation. In *Proc. IJCAI*, pages 287–294, 1969.

N. Murray and E. Rosenthal. Theory Links: Applications to Automated Theorem Proving. *J. of Symbolic Computation*, 4:173–190, 1987.

Robert Nieuwenhuis and Fernando Orejas. Clausal Rewriting. In S. Kaplan and M. Okada, editors, *Proceedings of the 2nd International Workshop on Conditional and Typed Rewriting Systems*, pages 246–258. Springer-Verlag, June 1990. LNCS 516.

Robert Nieuwenhuis and Albert Rubio. Theorem Proving with Ordering Constrained Clauses. In D. Kapur, editor, *Proceedings of the 11th International Conference on Automated Deduction (CADE-11)*, pages 477–491. Springer-Verlag, June 1992. LNAI 607.

H.-J. Ohlbach. Compilation of Recursive Two-Literal Clauses into Unification Algorithms. In V. Sgurev Ph. Jorrand, editor, *Artificial Intelligence IV – Methodology, Systems, Applications*. Norh Holland, 1990.

Etienne Paul. Equational Methods in First Order Predicate Calculus. *Journal of Symbolic Computation*, 1(1), March 1985.

E. Paul. On Solving the Equality Problem in Theories Defined by Horn Clauses. *Theoretical Computer Science*, 44:127–153, 1986.

U. Petermann. How to build in an open theory into connection calculi. *submitted to J. on Computers and Artificial Intelligence*, 1991.

David A. Plaisted, Geoff Alexander, Heng Chu, and Shie-Jue Lee. Conditional Term Rewriting and First-Order Theorem Proving, October 1993.

D. Plaisted. A Sequent-Style Model Elimination Strategy and a Positive Refinement. *Journal of Automated Reasoning*, 4(6):389–402, 1990.

G. A. Robinson and L. Wos. Paramodulation and Theorem Proving in First Order Theories with Equality. In Meltzer and Mitchie, editors, *Machine Intelligence 4*. Edinburg University Press, 1969.

Jörg H. Siekmann. Unification Theory. *Journal of Symbolic Computation*, 7(1):207–274, January 1989.

J. Steinbach. Improving Associative Path Orderings. In M.E. Stickel, editor, *Proc CADE 10, LNCS 449*, pages 411–425. Springer, 1990.

M.E. Stickel. Automated Deduction by Theory Resolution. *Journal of Automated Reasoning*, 1:333–355, 1985.
M. Stickel. An Introduction to Automated Deduction. In Bibel, Biermann, Delgrande, Huet, Jorrand, Shapiro, Mylopoulos, and Stickel, editors, *Fundamentals of Artificial Intelligence*, pages 75–132. Springer, 1986.
M. Stickel. A Prolog Technology Theorem Prover: A New Exposition and Implementation in Prolog. Technical note 464, SRI International, 1989.
M. Stickel. A Prolog Technology Theorem Prover. In M.E. Stickel, editor, *Proc CADE 10, LNCS 449*, pages 673–675. Springer, 1990.
M. Stickel. Upside-Down Meta-Interpretation of the Model Elimination Theorem-Proving Procedure for Deduction and Abduction, June 1991.
F. Stolzenburg and P. Baumgartner. Constraint Model Elimination and a PTTP-Implementation. Research Report 10/94, University of Koblenz, 1994.
G. Sutcliffe, C. Suttner, and T. Yemenis. The TPTP problem library. In *Proc. CADE-12*. Springer, 1994.
R. Veroff and L. Wos. The linked inference principle, i: the formal treatment. *Journal of Automated Reasoning*, 8(2), 1992.
H. Zhang and D. Kapur. First-Order Theorem Proving Using Conditional Rewrite Rules. In R. Overbeek E. Lusk, editor, *Proc. 9th CADE, LNCS 310*, pages 1–20. Springer, 1988.

Appendix

This appendix contains the missing proofs from the main part of this paper. Also, if an additional lemma is needed solely for a proof and has no further relevance outside the scope of the proof, it is given here.

SECTION 3

LEMMA 4.**(Ground completeness of $\mathcal{I}_0(\mathcal{T})$)** *Let $\mathcal{T}$ be a ground theory (i.e. a theory consisting of ground clauses only) and M be a set of ground literals. If M is $\mathcal{T}$-unsatisfiable then $L \Longrightarrow^*_{\mathcal{I}_0(\mathcal{T}),M\cup punit(\mathcal{I}_0(\mathcal{T}))}$ false for some $L \in M \cup punit(\mathcal{I}_0(\mathcal{T}))$.*

Proof. By definition of $\mathcal{T}$-unsatisfiability M is $\mathcal{T}$-unsatisfiable iff $M \cup \mathcal{T}$ is unsatisfiable, where M is considered as a set of unit clauses. By propositional compactness we may assume now that $M \cup \mathcal{T}$ is finite.

We apply induction on the size n of the atom set of $M \cup \mathcal{T}$ (the atom set of a Horn clause set consists of all atoms occurring in clauses in it).

Base case ($n = 1$): $M \cup \mathcal{T}$ must contain a positive literal A and a purely negative clause $\neg A^1 \vee \ldots \vee \neg A^k$ with $k \geq 1$ occurrences of $\neg A$ (the superscripts denote the distinct occurrences of the same literal $\neg A$).

The literal A and the clause $\neg A^1 \vee \ldots \vee \neg A^k$ may be contained in M or in $\mathcal{T}$, which results in the following cases:

If $k = 1$ then $\neg A \in \mathcal{T}$ or $\neg A \in M$. If $\neg A \in \mathcal{T}$ then the same refutation as in case $k > 1$ below exists. Therefore suppose now $\neg A \in M$. We have $A \in M$ or $A \in \mathcal{T}$. If $A \in M$ then a refutation

$$A \overset{\neg A}{\Longrightarrow}_{A,\neg A \rightarrow false} false$$

exists. If $A \in \mathcal{T}$ then by definition of $\mathcal{I}_0$, $\neg A \rightarrow \mathit{false} \in \mathcal{I}_0(\mathcal{T})$. In this case a refutation

$$\neg A \stackrel{\varepsilon}{\Longrightarrow}_{\neg A \rightarrow \mathit{false}} \mathit{false}$$

exists.

If $k > 1$ then $\neg A^1 \vee \ldots \vee \neg A^k \in \mathcal{T}$ follows. By definition of $\mathcal{I}_0$ $A^1 \cdots A^k \rightarrow \mathit{false} \in \mathcal{I}_0(\mathcal{T})$. We have $A \in M$ or $A \in \mathcal{T}$. However, since a theory is satisfiable by definition, $A \in \mathcal{T}$ is not possible. Hence $A \in M$. Thus a refutation

$$A^1 \stackrel{A^2 \cdots A^k}{\Longrightarrow}_{A^1, \ldots, A^k \rightarrow \mathit{false}} \mathit{false}$$

from $M \cup \mathit{punit}(\mathcal{I}_0(\mathcal{T}))$ exists.

Induction step ($n > 1$): $M \cup \mathcal{T}$ must contain at least one positive unit clause. (proof: if not, then every clause contains at least one negative literal. But then an interpretation that assigns *true* to every negative literal is a model for $M \cup \mathcal{T}$. Contradiction). Thus let $A \in M \cup \mathcal{T}$ be a positive unit clause. If $A \in M$ then clearly $A \in M \cup \mathit{punit}(\mathcal{I}_0(T))$. If $A \in \mathcal{T}$ then $\neg A \rightarrow \mathit{false} \in \mathcal{I}_0(\mathcal{T})$ and thus $A \in \mathit{punit}(\mathcal{I}_0(T))$. Thus always $A \in M \cup \mathit{punit}(\mathcal{I}_0(T))$. This fact will be used below.

If $M \cup \mathcal{T}$ contains a clause $\neg A \vee \ldots \vee \neg A$ the same argument as for $n = 1$ applies. This check guarantees that the following processing does not yield the empty clause.

Let $\mathcal{T}'$, resp. M', be obtained from $\mathcal{T}$, resp. M, by deleting every clause of the form $A \vee R$ (R may be empty), and then by replacing every clause of the form $\neg A^1 \vee \ldots \vee \neg A^k \vee R$ (where $\neg A$ does not occur in R, and R cannot be empty) by R. In $\mathcal{T}' \cup M'$ neither A nor $\neg A$ occurs. Thus the atom size of $M' \cup \mathcal{T}'$ is $n-1$. Furthermore $M' \cup \mathcal{T}'$ must be unsatisfiable, because otherwise a model for $M' \cup \mathcal{T}'$ can be extended to a model that assigns *true* to A, which would be in turn a model for $M \cup \mathcal{T}$, and thus M would be $\mathcal{T}$-satisfiable. Hence we can apply the induction hypothesis and assume that a $\mathcal{I}_0(\mathcal{T}')$-refutation of $M' \cup \mathit{punit}(\mathcal{I}_0(\mathcal{T}'))$ exists. Let D' be that refutation. We show how to transform D' into a $\mathcal{I}_0(\mathcal{T})$-refutation of $M \cup \mathit{punit}(\mathcal{I}_0(\mathcal{T}))$. For this, every derivation step involving a positive literal $B \in M' \cup \mathit{punit}(\mathcal{I}_0(\mathcal{T}'))$ which is not contained in $M \cup \mathit{punit}(\mathcal{I}_0(\mathcal{T}))$ has to be eliminated from D' (subcase 1), *and* every derivation step involving an inference rule $P \rightarrow C \in \mathcal{I}_0(\mathcal{T}')$ which is not contained in $\mathcal{I}_0(\mathcal{T})$ has to be eliminated from D' (subcase 1)[16].

Subcase 1: We conclude from $M' \subseteq M$ that $B \notin \mathit{punit}(\mathcal{I}_0(\mathcal{T}))$. From $B \in \mathit{punit}(\mathcal{I}_0(\mathcal{T}'))$ it follows $B \in \mathcal{T}'$. Since $B \notin \mathit{punit}(\mathcal{I}_0(\mathcal{T}))$ it follows $B \notin \mathcal{T}$. Hence $B \in \mathcal{T}'$ is obtained from $\neg A^1 \vee \ldots \vee \neg A^k \vee B \in \mathcal{T}$ by the replacement operation described above. With $A \in M \cup \mathit{punit}(\mathcal{I}_0(\mathcal{T}))$ and $A_1, \ldots, A_k \rightarrow B \in \mathcal{I}_0(\mathcal{T})$ a $\mathcal{I}_0(\mathcal{T})$-derivation

$$D_B = (A^1 \stackrel{A^2 \cdots A^k}{\Longrightarrow}_{A_1, \ldots, A_k \rightarrow B} B)$$

of B from $M \cup \mathit{punit}(\mathcal{I}_0(\mathcal{T}))$ exists. Hence a $\mathcal{I}_0(\mathcal{T}')$-derivation B from $M' \cup \mathit{punit}(\mathcal{I}_0(\mathcal{T}'))$ occurring in D' can be replaced by the $\mathcal{I}_0(\mathcal{T})$-derivation D_B from $M \cup \mathit{punit}(\mathcal{I}_0(\mathcal{T}))$.

[16] To be completely formal, this requires induction on the number of applications of the rule $P \rightarrow C$ and the literal B in D'

Subcase 2: We have $P \rightarrow C \in \mathcal{I}_0(\mathcal{T}')$ but $P \rightarrow C \notin \mathcal{I}_0(\mathcal{T})$. Here we distinguish two cases: in the first case, $P \rightarrow C$ is of the form $\neg B \rightarrow false$. Thus $B \in \mathcal{T}'$ was obtained from $\neg A^1 \vee \ldots \vee \neg A^k \vee B \in \mathcal{T}$ by the replacement operation above. Hence $A^1, \ldots, A^k \rightarrow B \in \mathcal{I}_0(\mathcal{T})$ and we can again find the $\mathcal{I}_0(\mathcal{T})$-derivation D_B of B from $M \cup punit(\mathcal{I}_0(\mathcal{T}))$. D' is of the form

$$D' = (L_1 \overset{D_1}{\Longrightarrow}_{R_1} \ldots L_n \overset{D_n}{\Longrightarrow}_{R_n} \neg B \overset{\varepsilon}{\Longrightarrow}_{\neg B \rightarrow false} false)$$

Now replace the last inference with $\neg B \rightarrow false$ by an inference with $\neg B, B \rightarrow false \in \mathcal{I}_0(\mathcal{T})$ to obtain the refutation

$$L_1 \overset{D_1}{\Longrightarrow}_{R_1} \ldots L_n \overset{D_n}{\Longrightarrow}_{R_n} \neg B \overset{D_B}{\Longrightarrow}_{\neg B, B \rightarrow false} false$$

In the second case $P \rightarrow C$ is not of the form $\neg B \rightarrow false$. By definition of $\mathcal{I}_0$, $P \rightarrow C$ then must be of the form $B_1, \ldots, B_l \rightarrow C$, where the B_is are positive literals and C is either a positive literal or *false* . Thus $\mathcal{T}'$ contains a clause $\neg B_1 \vee \ldots \vee \neg B_l \vee C$ or $\neg B_1 \vee \ldots \vee \neg B_l$, respectively. The further argumentation holds for both cases. Let us therefore consider only the first case. $\neg B_1 \vee \ldots \vee \neg B_l \vee C$ is obtained from the clause $\neg A^1 \vee \ldots \neg A^k \vee \neg B_1 \vee \ldots \vee \neg B_l \vee C \in \mathcal{T}$ by the replacement operation above. So $A^1, \ldots, A^k, B_1, \ldots, B_l \rightarrow C \in \mathcal{I}_0(\mathcal{T})$. Since $A \in M \cup punit(\mathcal{I}_0(\mathcal{T}))$ every derivation step

$$B_1 \overset{D_2 \cdots D_l}{\Longrightarrow}_{B_1, \ldots, B_l \rightarrow C} C$$

in D' with $B_1, \ldots, B_l \rightarrow C \in \mathcal{I}_0(\mathcal{T}')$ can be replaced by an inference with $A^1, \ldots, A^k, B_1, \ldots, B_l \rightarrow C \in \mathcal{I}_0(\mathcal{T})$ to obtain

$$B_1 \overset{A^1 \cdots A^k D_2 \cdots D_l}{\Longrightarrow}_{A^1, \ldots, A^k, B_1, \ldots, B_l \rightarrow C} C$$

SECTION 4

PROPOSITION 9. *The relation $\succ_{Lin}$ is a monotonic derivation ordering.*

Proof. By definition $D \succ_{Lin} E$ iff $compl(D) \gg_{NMW} compl(E)$. By Proposition 7, $\gg_{NMW}$ is a simplification ordering and hence by Theorem 6 well-founded.

Monotonicity is proven by simple structural induction on the derivation, using in the induction step the fact that $\gg_{NMW}$ is monotonic. More formally, let D be a derivation and suppose $D|_{p,i,j} = G$, F is a derivation which agrees with G on top literal and derived literal and $G \succ_{Lin} F$. We have to show $D \succ_{Lin} D[F]_{p,i,j}$.
Base case: if $p = \lambda$ then $D = D|_p$ is of the form

$$D = (L_1 \overset{D_1}{\Longrightarrow} L_2 \ldots \underbrace{L_i \overset{D_i}{\Longrightarrow} L_{i+1} \ldots L_{j-1} \overset{D_{j-1}}{\Longrightarrow} L_j}_{=G} \overset{D_j}{\Longrightarrow} L_{j+1} \ldots L_{n-1} \overset{D_{n-1}}{\Longrightarrow} L_n)$$

where $1 \leq i < j \leq n$ (the case $i = j$ is impossible since then G were a trivial derivation, whose complexity is a bottom element wrt. $\succ_{Lin}$). The complexities of G and D can be written as

$$\begin{aligned} compl(G) &= \{\!\{0\}\!\} \cup R_G \\ &\quad \text{where } R_G = \{\!\{compl(D_i)_{w_i}, \ldots, compl(D_{j-1})_{w_{j-1}}\}\!\} \qquad (2) \\ compl(D) &= \{\!\{0, compl(D_1)_{w_1}, \ldots, compl(D_{i-1})_{w_{i-1}}\}\!\} \cup R_G \\ &\quad \cup \{\!\{compl(D_j)_{w_j}, \ldots, compl(D_{n-1})_{w_{n-1}}\}\!\} \qquad (3) \end{aligned}$$

Concerning the derivation F, $compl(F)$ can be written as

$$compl(F) = \{\!\{0\}\!\} \cup R_F \qquad (4)$$

By definition, $G \succ_{Lin} F$ iff $compl(G) \succcurlyeq_{NMW} compl(F)$. But then it follows with (2) and (4) by monotonicity property (Theorem 6) of $\succcurlyeq_{NMW}$ (deleting identical elements does not change the relationship among multisets with weights) $R_G \succ_{NMW} R_F$ (*).

Now consider the derivation $D[F]_{\lambda,i,j}$; its complexity is

$$\begin{aligned} compl(D[F]_{\lambda,i,j}) &= \{\!\{0, compl(D_1)_{w_1}, \ldots, compl(D_{i-1})_{w_{i-1}}\}\!\} \cup \\ &\quad R_F \cup \{\!\{compl(D_j)_{w_j}, \ldots, compl(D_{n-1})_{w_{n-1}}\}\!\} \end{aligned}$$

From (*) it follows again by monotonicity property of $\succcurlyeq_{NMW}$ (replacing a subset by a smaller set makes the whole set smaller) $compl(D) \succcurlyeq_{NMW} compl(D[F]_{\lambda,i,j})$ and thus also $D \succ_{Lin} D[F]_{\lambda,i,j}$.

Induction step: p is of the form $k.l.r$ and thus D is of the form[17]

$$\begin{aligned} D &= (L_1 \overset{D_1}{\Longrightarrow} L_2 \ldots L_k \overset{D_k}{\Longrightarrow} L_{k+1} \ldots L_{n-1} \overset{D_{n-1}}{\Longrightarrow} L_n), \text{ where} \\ D_k &= D_k^1 \cdots D_k^l \cdots D_k^{n_k} \end{aligned}$$

Hence $compl(D)$ is of the form

$$\begin{aligned} compl(D) &= \{\!\{0, compl(D_1), \ldots, compl(D_{k-1}), \\ &\quad compl(D_k^1) \cup \cdots \cup compl(D_k^l) \cup \cdots \cup compl(D_k^{n_k}), \\ &\quad compl(D_{k+1}), \ldots, compl(D_n)\}\!\} \end{aligned}$$

with $D|_{k.l} = D_k^l$. In order to build $D[F]_{k.l.r,i,j}$ we have to replace D_k^l by $D_k^l[F]_r$. This means for the complexity of the new derivation

$$\begin{aligned} compl(D[F]_{k.l.r,i,j}) &= \{\!\{0, compl(D_1), \ldots, compl(D_{k-1}), \\ &\quad compl(D_k^1) \cup \cdots \cup compl(D_k^l[F]_r) \cup \cdots \cup compl(D_k^{n_k}), \\ &\quad compl(D_{k+1}), \ldots, compl(D_n)\}\!\} \end{aligned}$$

[17] Weights omitted.

By the induction hypothesis $D_k^l \succ_{Lin} D_k^l[F]_r$.
Thus by definition $compl(D_k^l) \not\succ_{NMW} compl(D_k^l[F]_r)$. Thus by monotonicity of $\succ$ $\succ_{NMW} compl(D) \succ_{NMW} compl(D[F]_{k.l.r,i,j})$ and hence $D \succ_{Lin} D[F]_{k.l.r,i,j}$.

The following lemma is needed in the proof of Proposition 11:

LEMMA 45. **(Instantiation Lemma for Linear Derivations)** *Suppose a linear $\mathcal{I}$-derivation*

$$D = (L_1 \overset{D_1}{\Longrightarrow} L_2 \ldots L_{n-1} \overset{D_{n-1}}{\Longrightarrow} L_n)$$

as given, and let γ be a ground substitution for L_1, M and L_n. Then there exists a linear ground $\mathcal{I}^g$-derivation

$$L_1\gamma \overset{D_1\gamma}{\Longrightarrow} L_2\gamma_2 \ldots L_{n-1}\gamma_{n-1} \overset{D_{n-1}\gamma}{\Longrightarrow} L_n\gamma$$

where γ_i is some ground substitution for L_i.

Thus, derivations may be ground instantiated; the additional (ground) substitutions γ_i come in due to extra variables in the conclusion of inference rules, and these variables have to be grounded in order to match the ground literal in the premise of the subsequent derivation step.

Proof.

Induction on the length n of the derivation.
Base case ($n = 1$): trivial, simply take $L_1\gamma$ as the desired derivation.
Induction step ($n - 1 \rightarrow n$): assume $n > 1$ and assume the result holds for derivations of length $n-1$. The concluding derivation step of the given derivation D is more precisely

$$L_{n-1} \overset{D_{n-1}}{\Longrightarrow}_{P_{n-1} \rightarrow C_{n-1},\sigma_{n-1}} L_n$$

where $\{\!|L_{n-1}|\!\} \cup D_{n-1} = P_{n-1}\sigma_{n-1}$ (*) and $C_{n-1}\sigma_{n-1} = L_n$.

Next consider the prefix $L_1 \overset{D_1}{\Longrightarrow} L_2 \ldots L_{n-2} \overset{D_{n-2}}{\Longrightarrow} L_{n-1}$ of D. Let γ' be a ground substitution for $L_{n-1}\gamma$. $\gamma\gamma'$ a ground substitution for L_1, M and L_{n-1} (because γ alone is a ground substitution, as given). Hence, by the induction hypothesis there exists a $\mathcal{I}^g$-derivation

$$D' = (L_1\gamma\gamma' \overset{D_1\gamma\gamma'}{\Longrightarrow} L_2\gamma_2 \ldots L_{n-2}\gamma_{n-2} \overset{D_{n-2}\gamma\gamma'}{\Longrightarrow} L_{n-1}\gamma\gamma')$$

Since $L_1\gamma$ being ground it follows $L_1\gamma = L_1\gamma\gamma'$. Similarly, since $M\gamma$ is ground and all literals in the sequence D_k are taken from M it follows that $D_k\gamma$ is also ground. But then $D_k\gamma = D_k\gamma\gamma'$. By these considerations D' is a $\mathcal{I}^g$-derivation of $L_{n-1}\gamma\gamma'$ from $M\gamma$ with top literal $L_1\gamma$.

With $P_{n-1}\sigma_{n-1} = \{\!|L_{n-1}|\!\} \cup D_{n-1}$, as given, it follows that D' can be extended by means of the substitution $\gamma\gamma'$ with one derivation step to a derivation

$$D'' = (D' \cdot (L_{n-1}\gamma\gamma' \overset{D_{n-1}\gamma\gamma'}{\Longrightarrow}_{(P_{n-1}\sigma_{n-1} \rightarrow C_{n-1}\sigma_{n-1})\gamma\gamma'} C_{n-1}\sigma_{n-1}\gamma\gamma'))$$

By the same arguments as for D_k above it holds $D_{n-1}\gamma = D_{n-1}\gamma\gamma'$; furthermore, with $L_n\gamma$ being ground and with $L_n \equiv C_{n-1}\sigma$ it follows that $L_n\gamma = C_{n-1}\sigma\gamma = C_{n-1}\sigma\gamma\gamma'$. Thus D'' is a derivation of $L_n\gamma$ from $M\gamma$.

We have to show that D'' is c $\mathcal{I}^g$-derivation, i.e. that the used inference rule in the last step is contained in $\mathcal{I}^g$. Since $D_{n-1}\gamma$ is ground (as concluded above) and $L_{n-1}\gamma\gamma'$ is ground (by definition of γ') and $C_{n-1}\sigma\gamma\gamma'$ is ground (because $C_{n-1}\sigma\gamma\gamma' = L_n\gamma$) it follows with (*) that $(P \to C)\gamma\gamma'$ is also ground. Hence from $P \to C \in \mathcal{I}$ it follows $(P \to C)\gamma\gamma' \in \mathcal{I}^g$, and so D'' is a $\mathcal{I}^g$-derivation. Setting $\gamma_{n-1} := \gamma\gamma'$ shows that D'' is the desired derivation.

PROPOSITION 11. **(Sufficient $\succ_{Lin}$-redundancy criterion)** *Let $\mathcal{I}$ be an inference system and $P \to_w C$ be an inference rule. Suppose that for every $L \in P$ there exists a linear $\mathcal{I} \setminus \{P \to_w C\}$-derivation from P*

$$L \equiv L_1 \overset{D_1}{\Longrightarrow}_{P_1 \to_{w_1} C_1} L_2 \overset{D_2}{\Longrightarrow}_{P_1 \to_{w_2} C_1} L_3 \cdots L_{n-1} \overset{D_{n-1}}{\Longrightarrow}_{P_1 \to_{w_{n-1}} C_1} L_n \equiv C$$

with $n \geq 1$ and such that for $i = 1, \ldots, n-1$ it holds $\langle P \setminus \{\!\{L\}\!\}, w\rangle \not\succ_{MW} \langle D_i, w_i\rangle$ In this comparison the sequence D_i of literals is to be read as a multiset. Then $P \to_w C$ is $\succ_{Lin}$-redundant in $\mathcal{I}$.

Proof. By contradiction. Suppose the assumptions of the proposition hold and $P \to C$ is not $\succ_{Lin}$-redundant in $\mathcal{I}$. Then by definition of $\succ_{Lin}$-redundancy (case 1) $P \to C$ is of the form $K \to \mathit{false}$, or (case 2) for some inference system $\mathcal{J} \supseteq \mathcal{I}$, some ground literal L_1, some ground literal set M and some ground literal L_n there exists a derivation $D = (L_1 \overset{*}{\Longrightarrow}_{(\mathcal{J} \cup \{P \to C\})^g, M} L_n)$ but there does not exist a derivation $D' = L_1 \overset{*}{\Longrightarrow}_{(\mathcal{J} \setminus \{P \to C\})^g, M} L_n$ with $D' \prec_{Lin} D$ (*).

Case 1: Suppose $P \to C$ is of the form $K \to \mathit{false}$ where K is a literal. By the assumption of the proposition we know there exists a derivation such that for every side literals D_i of the i-th derivation step, $D_i \subset P \setminus \{\!\{K\}\!\} = \emptyset$ which is impossible, or else the weight of the inference rule used in the i-th step is < 0, which is also impossible.

Case 2: We are given that at least one ground instance of $P \to_w C$ is used in some derivation step $G = D|_{p,i,i+1}$ in D. W.l.o.g assume that no further ground instance of $P \to_w C$ is used in G (by tracing into D and its subderivations always such a "bottommost" derivation can be located).

We will show that there exists a $(\mathcal{J} \setminus \{P \to_w C\})^g$-derivation F with $F \prec_{Lin} G$ and which does not use $P \to_w C$ and which can replace G in D, i.e. we build $D' = D[F]_{p,i,i+1}$. Since $\succ_{Lin}$ is monotonic (Proposition 9) it then holds $D' \prec_{Lin} D$. Since F does not use a ground instance of $P \to_w C$ the number of usages of ground instances of $P \to_w C$ in D' is 1 less than in D. Hence repeated application of this procedure terminates and thus yields a $(\mathcal{J} \setminus \{P \to_w C\})^g$-derivation. But then by transitivity of $\succ_{Lin}$ we obtain a contradiction to (*).

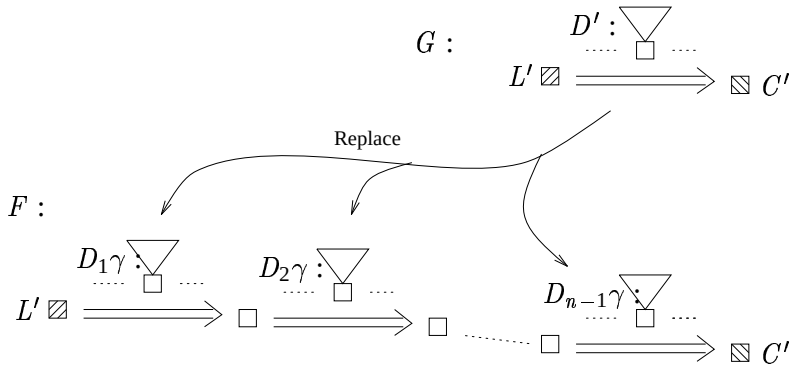

Fig. 13. Illustration of proof of Proposition 11.

Now for the construction of F (cf. Figure 13): the derivation step G can be written as

$$G = L' \overset{D'}{\Longrightarrow}_{(L, N \to C)\gamma} C'$$

where $P = \{\!|L|\!\} \cup N$, γ is a ground substitution, $L' = L\gamma$, $C' = C\gamma$ and D' is a sequence of derivations of $N\gamma$; the complexity is

$$compl(G) = \{\!|0, compl(D')|\!\}$$

From the assumption of the proposition we learn that there exists a $\mathcal{I} \setminus \{P \to_w C\}$-derivation

$$F' = (L \equiv L_1 \overset{D_1}{\Longrightarrow} \ldots \overset{D_{n-1}}{\Longrightarrow} L_n \equiv C)$$

such that $\langle D_i, w_i \rangle \prec\!\!\prec_{MW} \langle N, w \rangle$ (for $1 \leq i \leq n$), where w_i is the weight of the inference rule used in the i-th derivation step.

Applying the instantiation lemma to γ and F' (Lemma 45) yields a $(\mathcal{I} \setminus \{P \to C\})^g$-derivation

$$F'' = (L' \equiv L_1\gamma \overset{D_1\gamma}{\Longrightarrow} \ldots \overset{D_{n-1}\gamma}{\Longrightarrow} L_n\gamma = C')$$

Note that with $\mathcal{I} \subseteq \mathcal{J}$ it holds that F'' is a $(\mathcal{J} \setminus \{P \to C\})^g$-derivation as well.

From $\langle D_i, w_i \rangle \prec\!\!\prec_{MW} \langle N, w \rangle$ it follows

$$\langle D_i\gamma, w_i \rangle \prec\!\!\prec_{MW} \langle N\gamma, w \rangle \tag{5}$$

Since D' is a sequence of derivations of $N\gamma$ we can find for every sequence $D_i\gamma \subseteq N\gamma$ a sequence $D_i' \subseteq D'$ such that D_i' is a sequence of derivations of $D_i\gamma$. But then we can replace every side derivation $D_i\gamma$ in F'' by D_i', yielding still a $(\mathcal{J} \setminus \{P \to_w C\})^g$-derivation

$$F = (L' \overset{D_1'}{\Longrightarrow} \ldots \overset{D_n'}{\Longrightarrow} C')$$

with complexity

$$compl(F) = \{\!\{0, \langle compl(D'_1), w_1\rangle, \ldots, \langle compl(D'_{n-1}), w_{n-1}\rangle\}\!\}$$

Now consider (5) again: either it holds $D_i \subset N$ which implies by construction $D'_i \subseteq D'$, which in turn implies $\langle D'_i, w_i\rangle \prec\!\!\prec_{NMW} \langle D', w\rangle$; or else it holds $D_i = N$ and $w_i < w$, which implies by construction $D'_i = D'$ and hence also $\langle D'_i, w_i\rangle \prec\!\!\prec_{NMW} \langle D', w\rangle$. Thus in any case $F \prec Lin\,G$ which was to be shown. This concludes the proof of case 2.

SECTION 5

LEMMA 16. *Let $\mathcal{S}$ be a transformation system with derivation ordering $\succ$. Suppose $P \to C$ is $\succ$-redundant in some $\mathcal{I}$, and suppose that $\mathcal{I} \vdash_{\mathcal{S}} \mathcal{J}$. Then $P \to C$ is also $\succ$-redundant in $\mathcal{J}$.*

Proof. $\mathcal{J}$ is obtained from $\mathcal{I}$ by deletion of a $\succ$-redundant rule (case 1), or by adding a new rule (case 2).

Case 1: We have $\mathcal{J} = \mathcal{I} \setminus \{P' \to C'\}$ for some rule $P' \to C' \in \mathcal{I}$ to be deleted. By definition of redundancy we have to show that whenever

$$D = (L_1 \Longrightarrow^*_{((\mathcal{I}\setminus\{P'\to C'\})\cup\{P\to C\})^g, M} L_n) \tag{6}$$

and $P \to C$ is used in D then there exists a

$$D' = (L_1 \Longrightarrow^*_{((\mathcal{I}\setminus\{P'\to C'\})\setminus\{P\to C\})^g, M} L_n) \tag{7}$$

such that $D' \prec D$.

From $\mathcal{I} \setminus \{P' \to C'\} \subset \mathcal{I}$ and (6) it follows $D = (L_1 \Longrightarrow^*_{(\mathcal{I}\cup\{P\to C\})^g, M} L_n)$. We are given that $P \to C$is $\succ$-redundant in $\mathcal{I}$. Hence there exists a derivation $D' = L_1 \Longrightarrow^*_{(\mathcal{I}\setminus\{P\to C\})^g, M} L_n$ such that $D' \prec D$. Now, if no ground instance of $P' \to C'$ is used in D' then D' is also a $((\mathcal{I} \setminus \{P' \to C'\}) \setminus \{P \to C\})^g$-derivation, which proves (7). Otherwise, application of Lemma 46 to D' also proves (7).

Case 2: We have $\mathcal{J} = \mathcal{I} \cup \{P' \to C'\}$ for some new rule $P' \to C'$. By definition of redundancy we have to show that whenever

$$D = (L_1 \Longrightarrow^*_{((\mathcal{I}\cup\{P'\to C'\})\cup\{P\to C\})^g, M} L_n) \tag{8}$$

and $P \to C$ is used in that derivation then there exists a derivation

$$D' = (L_1 \Longrightarrow^*_{((\mathcal{I}\cup\{P'\to C'\})\setminus\{P\to C\})^g, M} L_n) \tag{9}$$

such that $D' \prec D$. Setting $\mathcal{J} := \mathcal{I} \cup \{P' \to C'\}$ in the definition of redundancy (Def. 10) renders this case trivial.

The following lemma is needed in the proof of Lemma 20.

LEMMA 46. *If*

1. $D = (L \Longrightarrow^*_{(\mathcal{I}\setminus\{P\to C\})^g} L')$, *where* $P \to C$ *is a* $\succ$*-redundant inference rule in* $\mathcal{I}$, *and*
2. *if in D some ground instance of* $P' \to C'$ *is used, and*
3. *if* $\mathcal{I} \vdash \mathcal{I} \setminus \{P' \to C'\} =: \mathcal{J}$ *by deletion*

then there exists a derivation

$$D' = (L \Longrightarrow^*_{(\mathcal{J}\setminus\{P\to C\})^g} L')$$

with $D' \prec D$.

Proof. Informally: if $P' \to C'$ is deleted in $\mathcal{J}$ then it must be redundant in $\mathcal{I}$. By redundancy the use of $P' \to C'$ in any derivation D' can be simulated by the remaining rules of $\mathcal{I}$. However, that simulating derivation, say D'' might use $P \to C$, which should not be used according to the lemma. On the other side, $P \to C$ itself is given as redundant in $\mathcal{I}$ and hence can be simulated by the remaining rules. However, that derivation, say D''' possibly contains usages of $P' \to C'$ again. Continuing this process will *not* fall into a loop due to strictly reduced complexity ($D''' \prec D'' \prec D'$) in every new derivation and well-foundedness of $\succ$.

Now the formal proof: if $P \to C \notin \mathcal{I}$ then the lemma holds trivially by the definition of $\succ$-redundancy.

Otherwise we apply well-founded induction on derivation orderings.

With $\mathcal{I} \setminus \{P' \to C'\} \subseteq \mathcal{I}$ it follows $D = (L \Longrightarrow^*_{\mathcal{I}^g} L')$. With $P' \to C'$ being deleted from $\mathcal{I}$, $P' \to C'$ must have been $\succ$-redundant in $\mathcal{I}$. We are given that a ground instance of $P' \to C'$ is used in D. Hence by definition of $\succ$-redundancy there exists a derivation

$$D' = L \Longrightarrow^*_{(\mathcal{I}\setminus\{P'\to C'\})^g} L' \tag{10}$$

with $D' \prec D$. Now we distinguish two cases:

Case 1: No ground instance of $P \to C$ is used in D'. But then by the existence of D' we find with (10) that $D' = L \Longrightarrow^*_{((\mathcal{I}\setminus\{P\to C\})\setminus\{P'\to C'\})^g} L'$, which proves the lemma.

Case 2: A ground instance of $P \to C$ is used in D'. First note that from $\mathcal{I} \setminus \{P' \to C'\} \subseteq \mathcal{I}$ by (10) it follows $D' = L \Longrightarrow^*_{\mathcal{I}^g} L'$. We are given that $P \to C$ is $\succ$-redundant in $\mathcal{I}$. By definition of $\succ$-redundancy we conclude that there exists a derivation $D'' = L \Longrightarrow^*_{(\mathcal{I}\setminus\{P\to C\})^g} L'$ such that $D'' \prec D' (\prec D)$. Now apply the induction hypothesis to D'' and conclude the result from transitivity of $\succ$.

LEMMA 20. *Let* $\mathcal{S}$ *be a transformation system with derivation ordering* $\succ$. *Let* $\mathcal{I}_0 \vdash \mathcal{I}_1 \vdash \cdots$ *be a* $\mathcal{S}$*-deduction. If for some k,* $P \to C$ *is* $\succ$*-redundant in* $\mathcal{I}_k$ *then* $P \to C$ *is* $\succ$*-redundant in* $\mathcal{I}_\infty$.

Proof. In order to prove the lemma suppose that

$$D = (L \Longrightarrow^*_{(\mathcal{I}_\infty\cup\{P\to C\})^g,M} L') \tag{11}$$

which uses a ground instance of $P \to C$. We have to show that there exists a derivation

$$D' = (L \Longrightarrow^*_{(\mathcal{I}_\infty \setminus \{P \to C\})^g, M} L') \tag{12}$$

with $D' \prec D$. As a consequence of Proposition 19 some $\mathcal{I}_m$ ($m \geq 0$) contains (at least) all those inference rules from $\mathcal{I}_\infty$, the ground instances of which are used in D. That is

$$D = (L \Longrightarrow^*_{(\mathcal{I}_m \cup \{P \to C\})^g, M} L') \tag{13}$$

We distinguish two cases:

Case 1: $m > k$. We are given that $P \to C$ is $\succ$-redundant in $\mathcal{I}_k$. Applying Lemma 16 $m - k$ times we conclude that $P \to C$ is $\succ$-redundant in $\mathcal{I}_m$ as well. Hence by definition of $\succ$-redundancy we find with (13) a derivation D_m

$$D_m = (L \Longrightarrow^*_{(\mathcal{I}_m \setminus \{P \to C\})^g, M} L') \tag{14}$$

with $D_m \prec D$.

Case 2: $m \leq k$. By Lemma 18 every inference rule in $\mathcal{I}_m$ is also contained in $\mathcal{I}_k$. Thus with (13) D is also $(\mathcal{I}_k \cup \{P \to C\})^g$-derivation. We are given that $P \to C$ is $\succ$-redundant in $\mathcal{I}_k$. Hence there exists a derivation $D_k = (L \Longrightarrow^*_{(\mathcal{I}_k \setminus \{P \to C\})^g, M} L'$ with $D_k \prec D$).

By taking $n = max(m, k)$ there exists in both cases a derivation

$$D_n = (L \Longrightarrow^*_{(\mathcal{I}_n \setminus \{P \to C\})^g, M} L')$$

with $D_n \prec D$.

We claim that for some $l \geq n$ there exists a derivation $D_l = L \Longrightarrow^*_{(\mathcal{I}_l \setminus \{P \to C\})^g, M} L'$ with $D_l \preceq D_n$, and the inference rules whose ground instances are used in D_l are never deleted afterwards, i.e. if $(P' \to C')\gamma \in (\mathcal{I}_l \setminus \{P \to C\})^g$ is a ground instance used in D_l then $P' \to C' \in \mathcal{I}_j \setminus \{P \to C\}$ for every $j \geq l$.

This claim then proves (12) and thus the lemma by transitivity of $\succ$ and by the following line of reasoning: suppose $P' \to C' \in \mathcal{I}_j \setminus \{P \to C\}$ for every $j \geq l$ then

$$\begin{aligned} P' \to C' \in \bigcap_{j \geq l} (\mathcal{I}_j \setminus \{P \to C\}) &= (\bigcap_{j \geq l} \mathcal{I}_j) \setminus \{P \to C\} \\ &\subseteq (\bigcup_{l \geq 0} \bigcap_{j \geq l} \mathcal{I}_j) \setminus \{P \to C\} \\ &= \mathcal{I}_\infty \setminus \{P \to C\} \end{aligned}$$

Hence if $(P' \to C')\gamma \in (\mathcal{I}_l \setminus \{P \to C\})^g$ is a used ground instance in D_l then also $(P' \to C')\gamma \in (\mathcal{I}_\infty \setminus \{P \to C\})^g$. Thus (12) follows.

It remains to prove the above claim. Starting with D_n we construct a sequence

$$D_n, D_{n+1}, D_{n+2}, \ldots$$

of derivations, where for $i > n$ we define

$$D_{i+1} := \begin{cases} D_i' & \text{if } \mathcal{I}_{i+1} \text{ is obtained from } \mathcal{I}_i \text{ by deletion of an inference rule } P'' \to C'', \text{ ground instances of which are used in } D_i, \text{ where } D_i' = (L \Longrightarrow^*_{(\mathcal{I}_{i+1}\setminus\{P\to C\})^g, M} L') \text{ and } D_i' \prec D_i. \text{ Such a derivation } D_i' \text{ exists because with } P \to C \text{ being } \succ\text{-redundant in } \mathcal{I}_n \text{ it follows by Lemma 16 that } P \to C \text{ is } \succ\text{-redundant in } \mathcal{I}_n, \mathcal{I}_{n+1}, \ldots, \mathcal{I}_i. \text{ Now apply Lemma 46.} \\ D_i & \text{else.} \end{cases}$$

Note that in this sequence, deleting an inference rule $P'' \to C''$ ground instances of which are used results in a strictly smaller derivation $D_i' \prec D_i$. Since $\succ$ is a derivation ordering and hence well-founded, we arrive at the chain $D_n, D_{n+1}, \ldots, D_l, D_l, D_l, \ldots$. Thus deletion of used inferences will not be continued infinitely. Stated positively, every inference system $\mathcal{I}_l, \mathcal{I}_{l+1}, \mathcal{I}_{l+2}$ contains every inference rule ground instances of which are used in D_l. Thus the claim above is proved, which concludes the proof of the lemma.

LEMMA 22. *Let $\mathcal{S}$ be a transformation system with derivation ordering $\succ$. Let $\mathcal{I}_0 \vdash \mathcal{I}_1 \vdash \cdots$ be a $\mathcal{S}$-deduction. If there exists a derivation $D = (L \Longrightarrow^*_{\mathcal{I}_k^g, M\cup punit(\mathcal{I}_k^g)} L')$ then there also exists a derivation $D' = (L \Longrightarrow^*_{\mathcal{I}_\infty^g, M\cup punit(\mathcal{I}_\infty^g)} L')$ with $D' \preceq D$.*

Proof. The proof is similar to that of Lemma 20 above. As a consequence of Lemma 21 every input literal from $M \cup punit(\mathcal{I}_k^g)$ is also contained in $M \cup punit(\mathcal{I}_\infty^g)$. Thus from the assumption of the lemma it follows $D = (L \Longrightarrow^*_{\mathcal{I}_k^g, M\cup punit(\mathcal{I}_\infty^g)} L')$. For ease of notation define $N := M \cup punit(\mathcal{I}_\infty^g)$.

We claim that for some $l \geq k$ there exists a $D_l = L \Longrightarrow^*_{\mathcal{I}_l^g, N} L'$ with $D_l \preceq D$, and the inference rules whose ground instances are used in D_l are never deleted afterwards, i.e. if $(P' \to C')\gamma \in \mathcal{I}_l^g$ is a ground instance used in D_l then $P' \to C' \in \mathcal{I}_j$ for every $j \geq l$.

This claim then proves the lemma by the following line of reasoning: suppose $P' \to C' \in \mathcal{I}_j$ for every $j \geq l$ then

$$P' \to C' \in \bigcap_{j\geq l} \mathcal{I}_j \subseteq \bigcup_{l\geq 0} \bigcap_{j\geq l} \mathcal{I}_j = \mathcal{I}_\infty$$

Hence if $(P' \to C')\gamma \in \mathcal{I}_l^g$ is a used ground instance in D_l then also $(P' \to C')\gamma \in \mathcal{I}_\infty^g$. Thus the lemma follows.

It remains to prove the above claim. Starting with $D_k := D$ we construct a sequence $D_k, D_{k+1}, D_{k+2}, \ldots$ of derivations, where for $i > k$ we define

$$D_{i+1} := \begin{cases} D_i' & \text{if } \mathcal{I}_{i+1} \text{ is obtained from } \mathcal{I}_i \text{ by deletion of an inference rule } P'' \to C'', \text{ ground instances of which are used in } D_i, \text{ where } D_i' = (L \Longrightarrow^*_{\mathcal{I}_{i+1}^g, M} L') \text{ and } D_i' \prec D_i. \text{ Such a derivation exists by definition of the deletion operation.} \\ D_i & \text{else.} \end{cases}$$

Note that in this sequence, deleting an inference rule $P'' \to C''$ ground instances of which are used results in a strictly smaller derivation $D_i' \prec D_i$. Since $\succ$ is a derivation ordering and hence well-founded, we arrive at the chain $D_n, D_{n+1}, \ldots, D_l, D_l, D_l, \ldots$. Thus deletion of used inferences will not be continued infinitely. Stated positively, every inference system $\mathcal{I}_l, \mathcal{I}_{l+1}, \mathcal{I}_{l+2}$ contains every inference rule ground instances of which are used in D_l. Thus the claim above is proved, which concludes the proof of the lemma.

SECTION 7

PROPOSITION 27. *The transformation system Lin is order-normalizing wrt. LinG.*

Proof. Let $\mathcal{I}$ be an inference system and D be a ground $\mathcal{I}^g$ refutation of M which is not linear, i.e. $D \notin LinG$. We have to show that there exists a $(\mathcal{I}')^g$-derivation $D' \prec_{Lin} D$ of M, where $\mathcal{I}' = \mathcal{I}$ or $\mathcal{I}'$ is obtained from $\mathcal{I}$ by application of some mandatory transformation rule.

Since D is given as a non-linear derivation at least one side derivation is not a sequence of trivial derivations. Thus D can be written as

$$D = (L_1 \overset{D_1}{\Longrightarrow} L_2 \ldots \overset{D_{i-1}}{\Longrightarrow} \underbrace{L_i \overset{D_i}{\Longrightarrow} L_{i+1}}_{=:D''} \overset{D_{i+1}}{\Longrightarrow} \ldots L_{n-1} \overset{D_{n-1}}{\Longrightarrow} L_n)$$

where D_i is that critical side derivation and $D'' := D|_{\lambda,i,i+1}$. More precisely, the sub-derivation D'' is of the form

$$D'' = (L_i \overset{D^{K\gamma} D^{M\gamma}}{\Longrightarrow}_{(L,K,M \to L')\gamma} L_{i+1})$$

where $L_i = L\gamma$, $D_i = D^{K\gamma} D^{M\gamma}$, $D^{K\gamma}$ is a non-trivial derivation of $K\gamma$, $D^{M\gamma}$ is a sequence of derivations of $M\gamma$ and $(L, K, M \to L')\gamma \in \mathcal{I}^g$ is a ground instance of $L, K, M \to L' \in \mathcal{I}$ (cf. Figure 14). We will show that there exists a $(\mathcal{I}')^g$-derivation $D''' \prec_{Lin} D''$, where $\mathcal{I}' = \mathcal{I}$ or $\mathcal{I}'$ is obtained from $\mathcal{I}$ by application of a mandatory transformation rule from Lin. Then we define $D' := D[D''']_{\lambda,i,i+1}$. By monotonicity of $\succ_{Lin}$ (Proposition 9) then it follows $D' \prec_{Lin} D$, which was to be shown.

In general, the derivation $D^{K\gamma}$ is of the form

$$D^{K\gamma} = (\underbrace{J_1 \overset{E_1}{\Longrightarrow} J_2 \ldots J_{m-1} \overset{E_{m-1}}{\Longrightarrow} J_m}_{=:D^{J\delta}} \overset{D^{N\delta}}{\Longrightarrow}_{(J,N \to J')\delta} J'\delta)$$

where $m \geq 1$, $J_m = J\delta$, $D^{N\delta}$ is a (possibly empty) sequence of derivations of $N\delta$, $J'\delta = K\gamma$ and $(J, N \to J')\delta \in \mathcal{I}^g$ is a ground instance of $J, N \to J' \in \mathcal{I}$. Without loss of generality suppose that $J, N \to J'$ is variable disjoint from $L, K, M \to L'$. Consequently we may assume that the domains of δ and γ are disjoint, too. Hence $(J, N \to J')\delta\gamma = (J, N \to J')\delta$ and $(L, K, M \to L')\delta\gamma = (L, K, M \to L')\gamma$.

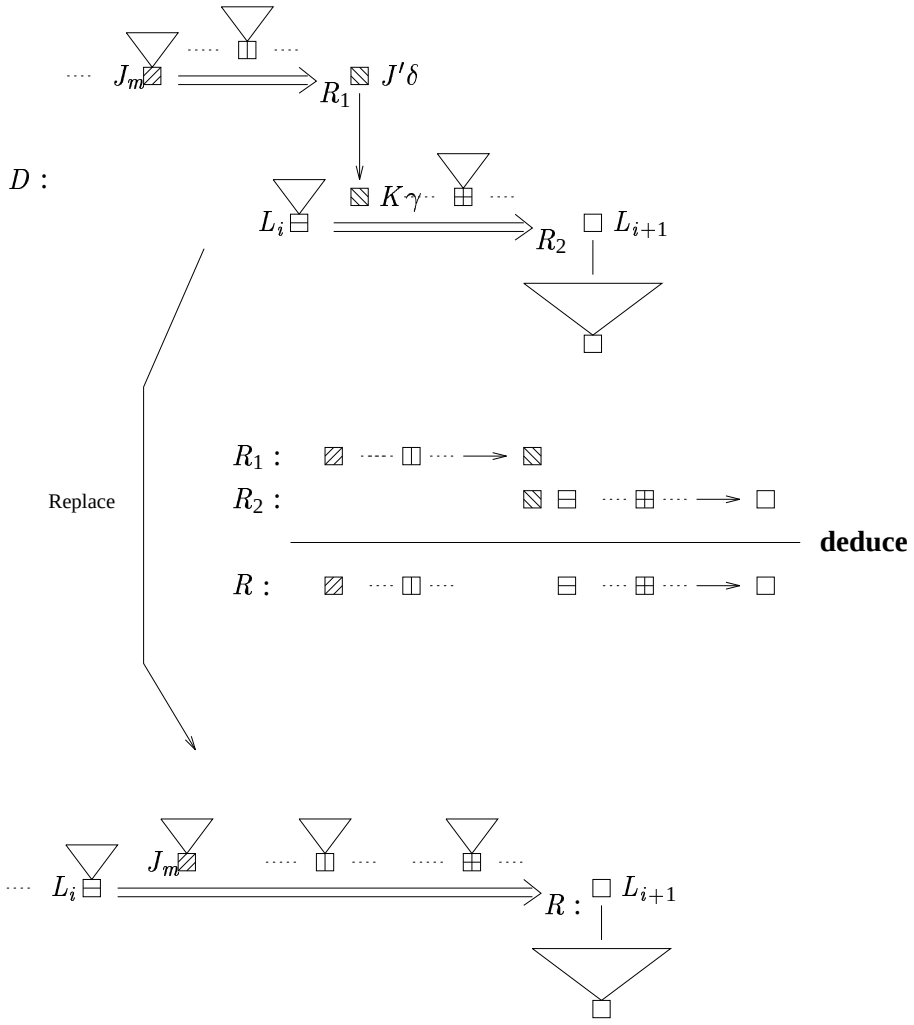

Fig. 14. Illustration of proof of Proposition 27 (ground case).

Together with with $J'\delta = K\gamma$ it follows that $J'\delta\gamma = K\delta\gamma$. In other words, $\delta\gamma$ is a unifier. Hence there exists a MGU σ and a substitution ϕ such that $\delta\gamma = \sigma\phi|_{dom(\delta\gamma)}$. By the existence of this MGU, the mandatory **Deduce** transformation rule can be applied to $L, K, M \rightarrow L'$ and $J, N \rightarrow J'$ by unifying J' and K with σ. The result is the inference rule $R := (L, J, N, M \rightarrow L')\sigma$.

Now either a variant of R already is contained in $\mathcal{I}$, and in this case define $\mathcal{I}' := \mathcal{I}$. Otherwise define $\mathcal{I}' := \mathcal{I} \cup \{R\}$. Since $R\phi$ is ground and $D^{J\delta}$, $D^{N\delta}$ and $D^{M\gamma}$ are appropriate ground $\mathcal{I}^g$-derivations (and hence also $(\mathcal{I}')^g$-derivations) there exists the

$(\mathcal{I}')^g$-derivation

$$D''' \;=\; (L_i \overset{D^{J\delta}D^{N\delta}D^{M\gamma}}{\Longrightarrow\!\!\!\!\!\!\!\!\!\!\!\succ}_{R\phi}\; L_{i+1})$$

Note that D''' and D'' coincide in top and derived literals. Hence the replacement as suggested above can be done. It remains to show $D''' \prec_{Lin} D''$. By definition of $\succ_{Lin}$ this is the same as to show $compl(D''') \prec_{NMW} compl(D'')$; for this proof the weights of the involved inference rules can be neglected, since decreasingness follows alone from properties of multiset orderings:

$$\begin{array}{rcl}
compl(D'') & = & \{\!\{0, compl(\{\!\{D^{K\gamma}\}\!\} \cup D^{M\gamma})\}\!\} \\
& = & \{\!\{0, compl(D^{K\gamma}) \cup compl(D^{M\gamma})\}\!\} \\
& = & \{\!\{0, \{\!\{0, compl(E_1), \ldots, compl(E_{m-1}), compl(D^{N\gamma})\}\!\} \\
& & \quad \cup compl(D^{M\gamma})\}\!\} \\
& = & \{\!\{0, compl(D^{J\delta}) \cup \{\!\{compl(D^{N\gamma})\}\!\} \cup compl(D^{M\gamma})\}\!\} \\
& \succ_{NMW} & \{\!\{0, compl(D^{J\delta}) \cup compl(D^{N\gamma}) \cup compl(D^{M\gamma})\}\!\} \\
& = & \{\!\{0, compl(D^{J\delta}D^{N\gamma}D^{M\gamma})\}\!\} \\
& = & compl(D''')
\end{array}$$

The transition $\succ_{NMW}$ is justified by property of nested multiset ordering (replacing a multiset by true subsets is decreasing).

This concludes the proof.

PROPOSITION 30. *The transformation system Lin is punit-normalizing wrt. LinG.*

Proof. Let $\mathcal{I}$ be an inference system and let

$$D = (T_1 \overset{D_1}{\Longrightarrow} T_2 \ldots T_n \overset{D_n}{\Longrightarrow} T_{n+1} \overset{D_{n+1}}{\Longrightarrow} T_{n+2})$$

be the given non-trivial linear ground $\mathcal{I}^g$ derivation from $M \cup punit(\mathcal{I}^g)$ such that $used(D) \cap punit(\mathcal{I}^g) \neq \emptyset$. We have to show that there exists a $(\mathcal{I}')^g$-derivation $D' \prec_{Lin} D$ of $M \cup punit((\mathcal{I}')^g)$, where $\mathcal{I}' = \mathcal{I}$ or $\mathcal{I}'$ is obtained from $\mathcal{I}$ by application of some mandatory transformation rule, and the top literal of D' is T_1 or a literal from $M \cup punit((\mathcal{I}')^g)$. Since D is given as non-trivial it holds $n \geq 0$.

Let $L\gamma \in used(D) \cap punit(\mathcal{I}^g)$ where $L\gamma$ is a ground instance of a literal $L \in punit(\mathcal{I})$. Two cases apply: (1) $L\gamma$ occurs in some D_i ($i \in \{1 \ldots n+1\}$) or (2) $L\gamma = T_1$.

Case 1: (cf. Figure 15) D contains a derivation step of the form

$$D'' := D|_{\lambda,i,i+1} = (T_i \overset{L\gamma E_1 \cdots E_{m_i}}{\Longrightarrow\!\!\!\!\!\!\!\!\!\!\!\succ}_{(T_i', L', E_1', \ldots, E_{m_i}' \to C)\gamma'}\; C\gamma') \qquad (*)$$

where $(T_i', L', E_1', \ldots, E_{m_i}' \to C)\gamma' \in \mathcal{I}^g$ is a ground instance of the rule $T_i', L', E_1', \ldots, E_{m_i}' \to C \in \mathcal{I}$, $T_i'\gamma' = T_i$, $L'\gamma' = L\gamma$, $E_j'\gamma' = E_j$ (for $j = 1 \ldots m_i$) and $C\gamma' \equiv T_{i+1}$ or $C\gamma' \equiv false$. The further reasoning holds for both cases. We will show

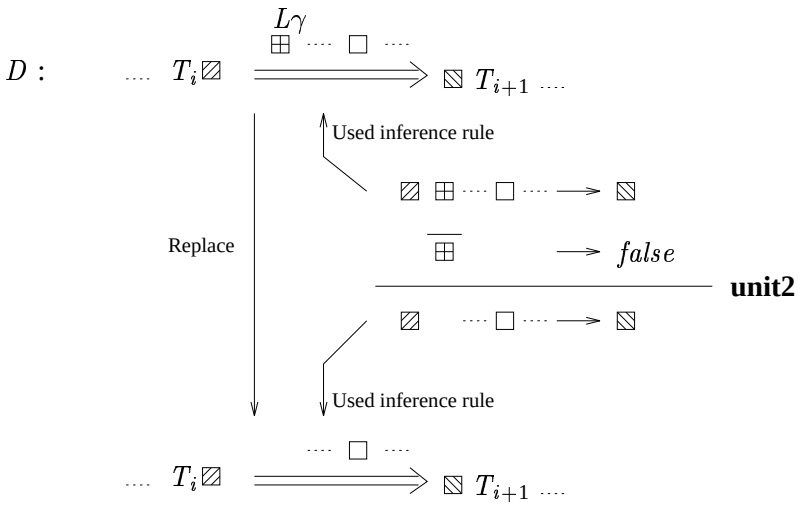

Fig. 15. A case in the proof of Proposition 30 (ground case).

how to deduce a new inference rule by which the application of $L\gamma$ in D'' can be omitted.

We will show that there exists a $(\mathcal{I}')^g$-derivation $D''' \prec_{Lin} D''$, where $\mathcal{I}' = \mathcal{I}$ or $\mathcal{I}'$ is obtained from $\mathcal{I}$ by application of a mandatory transformation rule from Lin. Then we define $D' := D[D''']_{\lambda,i,i+1}$. By monotonicity of $\succ_{Lin}$ (Proposition 9) then it follows $D' \prec_{Lin} D$, which was to be shown.

Since $L \in punit(\mathcal{I})$ by definition $\overline{L} \rightarrow false \in \mathcal{I}$. Let $\overline{L''} \rightarrow false$ be a new variant, variable disjoint from the premise $\{\!| T_i', L', E_1', \ldots, E_{m_i}' |\!\}$ of the applied inference rule. Since $L\gamma = L'\gamma'$ and L'' is a variant of L there exists a ground substitution γ'' with $L''\gamma'' = L'\gamma'$. Since L'' is a new variant, γ' can be supposed not to act upon the variables of L''. Hence $L''\gamma'\gamma'' = L'\gamma'\gamma''$. Since $\gamma'\gamma''$ is a unifier for L'' and L' there exists a most general unifier σ for L'' and L' and substitution δ such that

$$\sigma\delta|_{var(L')\cup var(L'')} = \gamma'\gamma''|_{var(L')\cup var(L'')} \quad (**)$$

By the existence of this MGU, the mandatory **Unit2** transformation rule can be applied to $T_i', L', E_1', \ldots, E_{m_i}' \rightarrow C \in \mathcal{I}$ and $\overline{L''} \rightarrow false \in \mathcal{I}$. The result is the new inference rule

$$R := (T_i', E_1', \ldots, E_{m_i}' \rightarrow C)\sigma$$

Now either a variant of R already is contained in $\mathcal{I}$, and in this case define $\mathcal{I}' := \mathcal{I}$. Otherwise define $\mathcal{I}' := \mathcal{I} \cup \{R\}$.

Let δ' be the restriction of γ' to the domain $var(\{\!| T_i', E_1', \ldots, E_{m_i}', C |\!\}) \setminus var(L')$. Together with (**) it follows

$$\{\!| T_i', E_1', \ldots, E_{m_i}', C |\!\}\sigma\delta\delta' = \{\!| T_i', E_1', \ldots, E_{m_i}', C |\!\}\gamma'\gamma'' = \{\!| T_i', E_1', \ldots, E_{m_i}', C |\!\}\gamma' \ .$$

By these equalities we can build the desired derivation

$$D''' := (T_i \overset{E_1 \cdots E_{m_i}}{\Longrightarrow}_{R\delta\delta'} C\gamma')$$

with rule $R\delta\delta' \in \mathcal{I}^g$.

Note that D''' and D'' coincide in top and derived literals. Hence the replacement as suggested above can be done. It remains to show $D''' \prec_{Lin} D''$. By definition of $\succ_{Lin}$ this is the same as to show $compl(D''') \prec_{NMW} compl(D'')$; for this proof the weights of the involved inference rules can be neglected, since decreasingness follows alone from properties of multiset orderings:

$$\begin{array}{rcl} compl(D'') & = & \{\!\{0, compl(L\gamma E_1 \cdots E_{m_i})\}\!\} \\ & = & \{\!\{0, \{\!\{0\}\!\} \cup compl(E_1 \cdots E_{m_i})\}\!\} \\ & \succ_{NMW} & \{\!\{0, compl(E_1 \cdots E_{m_i})\}\!\} \\ & = & compl(D''') \end{array}$$

Case 2: ($L\gamma = T_1$). We distinguish the cases (2.1) $D_1 \neq \varepsilon$ (i.e. D_1 is not the empty sequence of derivations) and (2.2) $D_1 = \varepsilon$.

Case 2.1: D begins with the derivation step

$$D'' = (L\gamma \overset{KE_1 \cdots E_{m_i}}{\Longrightarrow}_{(K', L', E'_1, \ldots, E'_{m_i} \to C)\gamma'} C\gamma') \qquad (*)$$

where $(L', K', E'_1, \ldots, E'_{m_i})\gamma' \to C\gamma' \in \mathcal{I}^g$ is a ground instance of the rule $L', K', E'_1, \ldots, E'_{m_i} \to C \in \mathcal{I}$, $L'\gamma' = L\gamma$, $K'\gamma' = K$, $E'_j\gamma' = E_j$ (for $j = 1 \ldots m_i$) and $C\gamma' \equiv T_2$ or $C\gamma' \equiv false$. The further reasoning holds for both cases. It is in close analogy to the case $L\gamma \in D_i$.

We will show that there exists a $(\mathcal{I}')^g$-derivation $D''' \prec_{Lin} D''$, where $\mathcal{I}' = \mathcal{I}$ or $\mathcal{I}'$ is obtained from $\mathcal{I}$ by application of a mandatory transformation rule from Lin. Then we define $D' := D[D''']_{\lambda,1,2}$. By monotonicity of $\succ_{Lin}$ (Proposition 9) then it follows $D' \prec_{Lin} D$, which was to be shown.

Since $L \in punit(\mathcal{I})$ by definition $\overline{L} \to false \in \mathcal{I}$. Let $\overline{L''} \to false$ be a new variant. As in case 1, the mandatory **Unit2** transformation rule can be applied to $\overline{L''} \to false$ and $L', K', E'_1, \ldots, E'_{m_i} \to C \in \mathcal{I}$, yielding

$$R := (K', E'_1, \ldots, E'_{m_i} \to C)\sigma$$

where σ is the MGU for L'' and L' used in that deduction step. Now either a variant of R already is contained in $\mathcal{I}$, and in this case define $\mathcal{I}' := \mathcal{I}$. Otherwise define $\mathcal{I}' := \mathcal{I} \cup \{R\}$.

As in case 1 there exists ground substitutions $\delta\delta'$, γ' and γ'' such that

$$\{\!\{K', E'_1, \ldots, E'_{m_i}, C\}\!\}\sigma\delta\delta' = \{\!\{K', E'_1, \ldots, E'_{m_i}, C\}\!\}\gamma'\gamma'' = \{\!\{K', E'_1, \ldots, E'_{m_i}, C\}\!\}\gamma'$$

By these equalities we can build the desired derivation

$$D''' := (K \overset{E_1 \cdots E_{m_i}}{\Longrightarrow}_{R\delta\delta'} C\gamma')$$

with rule $R\delta\delta' \in \mathcal{I}^g$.

Note that D''' and D'' coincide in top and derived literals. Hence the replacement as suggested above can be done. It remains to show $D''' \prec_{Lin} D''$. This proof is literally the same as the corresponding proof in case 1 above, except that $L\gamma$ is to be replaced by K. This completes the proof for the case $D_1 \neq \varepsilon$.

Case 2.2: ($D_1 = \varepsilon$). It holds that $n > 0$. Proof: Suppose, to the contrary that $n = 0$. Then the refutation consists of a single derivation step with the rule $T_1 \rightarrow \mathit{false} \in \mathcal{I}^g$. On the other side from $T_1 \in \mathit{punit}(\mathcal{I}^g)$ it follows $\overline{T_1} \rightarrow \mathit{false} \in \mathcal{I}^g$. But then $\mathcal{I}$ were not consistent, since no interpretation can satisfy both, T_1 and $\overline{T_1}$. However $\mathcal{I}$ is given as consistent. Contradiction.

The given derivation D can be written more specifically as ($n \geq 0$, cf. Figure 16) :

$$D = (L\gamma \overset{\varepsilon}{\Longrightarrow}_{(L' \rightarrow T_2')\gamma'} T_2 \ldots T_n \overset{D_n}{\Longrightarrow} T_{n+1} \overset{D_{n+1}}{\Longrightarrow} T_{n+2}) \qquad (*)$$

where $T_2 \neq \mathit{false}$, $(L' \rightarrow T_2')\gamma' \in \mathcal{I}^g$ is a ground instance of the rule $L' \rightarrow T_2 \in \mathcal{I}$, $L'\gamma' = L\gamma$ and $T_2'\gamma' = T_2$.

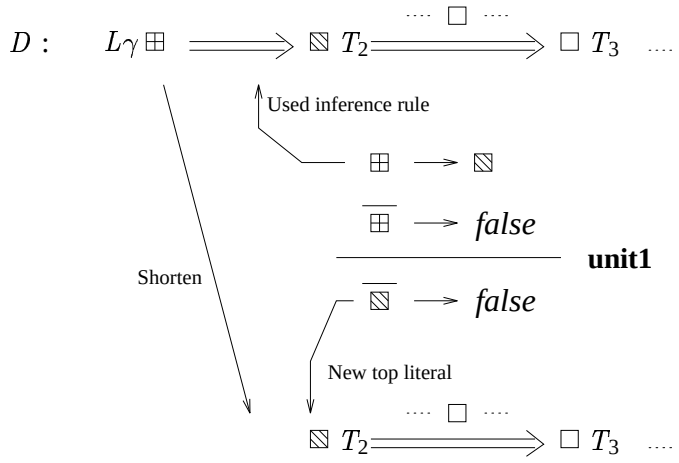

Fig. 16. A case in the proof of Lemma 30 (ground case)

Since $L \in \mathit{punit}(\mathcal{I})$ by definition $\overline{L} \rightarrow \mathit{false} \in \mathcal{I}$. Let $\overline{L''} \rightarrow \mathit{false}$ be a new variant, variable disjoint from the premise $\{L'\}$ of the applied inference rule. Since $L\gamma = L'\gamma'$ and L'' is a variant of L there exists a ground substitution γ'' with $L''\gamma'' = L'\gamma'$. Since L'' is a new variant, γ' can be supposed not to act upon the variables of L''. Hence

$L''\gamma'\gamma'' = L'\gamma'\gamma''$. Since $\gamma'\gamma''$ is a unifier for L'' and L' there exists a most general unifier σ for L'' and L' and substitution δ such that

$$\sigma\delta|_{var(L')\cup var(L'')} = \gamma'\gamma''|_{var(L')\cup var(L'')} \quad (**)$$

By the existence of this MGU, the mandatory **Unit1** transformation rule can be applied to $L' \to T'_2 \in \mathcal{I}$ and $\overline{L''} \to false \in \mathcal{I}$. The result is the new inference rule

$$R \;:=\; T'_2\sigma' \to false$$

Now either a variant of R already is contained in $\mathcal{I}$, and in this case define $\mathcal{I}' := \mathcal{I}$. Otherwise define $\mathcal{I}' := \mathcal{I} \cup \{R\}$.

Now let δ' be the restriction of γ' to the domain $var(T'_2) \setminus var(L')$. Together with (**) it follows

$$T'_2\sigma\delta\delta' = T'_2\gamma'\gamma''(= T'_2\gamma' = T_2) \quad (***)$$

Thus with $R \in \mathcal{I}'$ it follows $T_2 \in punit((\mathcal{I}')^g)$. By this fact we can cut off the first derivation step in D, yielding

$$D' := D|_{2,n+2} = (T_2 \ldots T_n \overset{D_n}{\Longrightarrow} T_{n+1} \overset{D_{n+1}}{\Longrightarrow} T_{n+2})$$

which is a derivation from $M \cup punit((\mathcal{I}')^g)$ as desired.

It remains to prove $D' \prec_{Lin} D$. By definition of $\succ_{Lin}$ this is the same as to show $compl(D') \prec_{NMW} compl(D)$; for this proof the weights of the involved inference rules can be neglected, since decreasingness follows alone from properties of multiset orderings:

$$\begin{array}{rcl} compl(D) & = & \{\!\{0, compl(\varepsilon), compl(D_2), \ldots, compl(D_{n+1})\}\!\} \\ & = & \{\!\{0, \{\}, compl(D_2), \ldots, compl(D_{n+1})\}\!\} \\ & \succ_{NMW} & \{\!\{0, compl(D_2), \ldots, compl(D_{n+1})\}\!\} \\ & = & compl(D') \end{array}$$

By concluding this final case the lemma is proven.

PROPOSITION 31. *Let $\mathcal{S}$ be a punit-normalizing transformation system wrt. $\mathcal{N}$, and let $\mathcal{I}$ be a completed inference system wrt. $\mathcal{S}$. Whenever there exists a ground derivation $D = (L \Longrightarrow^*_{\mathcal{I}^g, M\cup punit(\mathcal{I}^g)} L')$ with $D \in \mathcal{N}$ then there also exists a ground derivation $D' = (K \Longrightarrow^*_{\mathcal{I}^g, M} L')$ with $D' \preceq D$, $D' \in \mathcal{N}$ and some $K \in M$.*

Proof. By Noetherian induction on derivations wrt. the well-founded derivation ordering associated to $\mathcal{S}$. If $used(D) \cap punit(\mathcal{I}^g) = \emptyset$ then no literal from $punit(\mathcal{I}^g)$ is used in D then D is also a derivation from M alone. Hence we take $D' = D$.

Otherwise, by definition of punit-normalizing transformation systems there exists a derivation $D'' = (L \Longrightarrow^*_{(\mathcal{I}'')^g, M\cup punit((\mathcal{I}'')^g)} L')$ with $D'' \prec D$, where (1) $\mathcal{I}'' = \mathcal{I}$ or (2) $\mathcal{I} \vdash_{\mathcal{S}} \mathcal{I} \cup \{P \to C\} = \mathcal{I}''$ by some mandatory transformation rule from $\mathcal{S}$.

In case 1 D'' is also a $\mathcal{I}^g$-derivation of $M \cup punit(\mathcal{I}^g)$; now consider case 2: since $\mathcal{I}$ is completed it holds by Definition 23 $P \to C \in \mathcal{I}$ or (2.2) $P \to C$ is $\succ$-redundant in $\mathcal{I}$. In case 2.1 $\mathcal{I}'' = \mathcal{I}$ and hence D'' is also a $\mathcal{I}^g$-derivation of $M \cup punit(\mathcal{I}^g)$. In case 2.2 $P \to C$ is not of the form $L \to false$ (such rules are by definition never redundant). Hence $punit(\mathcal{I}'') = punit(\mathcal{I})$. If no ground instance of $P \to C$ is used in D'' then D'' is also a $\mathcal{I}^g$-derivation of $M \cup punit(\mathcal{I}^g)$, otherwise by definition of redundancy there exists a ground derivation $D''' = (L \Longrightarrow^*_{\mathcal{I}^g, M \cup punit((\mathcal{I})^g)} L')$ with $D''' \prec D''$ (note that this is a $\mathcal{I}^g$-derivation). From $D''' \prec D'' \prec D$ it follows by downward closure of $\mathcal{N}$ also $D'' \in \mathcal{N}$ and $D''' \in \mathcal{N}$. Thus, in any case there exists a $\mathcal{I}^g$-derivation of $M \cup punit(\mathcal{I}^g)$ which is contained in $\mathcal{N}$ and which is strictly smaller wrt. $\succ$ than the given derivation.

Now apply the induction hypothesis to that derivation.

SECTION 8

LEMMA 38. **(Top literal lemma)***Let $\mathcal{I}$ be a completed inference system wrt. the transformation system Lin. Suppose there exists a linear ground derivation $D = (L_1 \Longrightarrow^*_{\mathcal{I}^g, M} L_n \in LinG)$ with $L_1 \in M$. Let $T \in M$ such that $T \in used(M)$. Then there exists a linear ground derivation $D' = (T \Longrightarrow^*_{\mathcal{I}^g, M} L_n)$.*

Proof. Let the given derivation be

$$D = (L_1 \overset{M_1}{\Longrightarrow} L_2 \ldots \overset{M_{k-1}}{\Longrightarrow} L_k \overset{T\ M_k}{\Longrightarrow}_{R_1\gamma_1} L_{k+1} \overset{M_{k+1}}{\Longrightarrow} \ldots L_{n-1} \overset{M_{n-1}}{\Longrightarrow} L_n) \qquad (15)$$

where $n > 1$ (otherwise the claim is trivial), $k \in \{1 \ldots n\}$ and $R_1\gamma_1$ is a ground instance of $R_1 = L'_k, T', M'_k \to L'_{k+1} \in \mathcal{I}$, $L_k = L'_k\gamma_1$, $T = T'\gamma_1$, $M_k = M'_k\gamma_1$ and $L_{k+1} = L'_{k+1}\gamma_1$.

We do induction on the *top distance* k of the derivation step using T.

Base case: If $k = 1$ then the first derivation step is $L_1 \overset{T\ M_1}{\Longrightarrow}_{R_1\gamma_1} L_2$. By swapping T and L_1 it can be replaced by the derivation step $T \overset{L_1\ M_1}{\Longrightarrow}_{R_1\gamma_1} L_2$ which yields the desired linear derivation.

Induction step: For the induction step suppose that $k > 1$ and the claim to hold for derivations with top distance strictly smaller than k. D then can be written as

$$\underbrace{L_1 \overset{M_1}{\Longrightarrow} L_2 \ldots \overset{M_{k-2}}{\Longrightarrow} L_{k-1}}_{D_1} \overbrace{\overset{M_{k-1}}{\Longrightarrow}_{R_2\gamma_2} L_k \overset{T\ M_k}{\Longrightarrow}_{R_1\gamma_1} L_{k+1}}^{D_2} \underbrace{\overset{M_{k+1}}{\Longrightarrow} \ldots \overset{M_{n-1}}{\Longrightarrow} L_n}_{D_3} \qquad (16)$$

where $n \geq 3$ and $R_2\gamma_2$ is a ground instance of $R_2 = L'_{k-1}, M'_{k-1} \to L'_k \in \mathcal{I}$, $L_{k-1} = L'_{k-1}\gamma_2$, $M_{k-1} = M'_{k-1}\gamma_2$ and $L_k = L'_k\gamma_2$ (cf. Figure 17). Without loss of generality suppose that R_1 is variable disjoint from R_2. Consequently we may assume that the domains of γ_1 and γ_2 are disjoint, too. Hence $R_1\gamma_1\gamma_2 = R_1\gamma_1$ and $R_2\gamma_1\gamma_2 = R_2\gamma_2$.

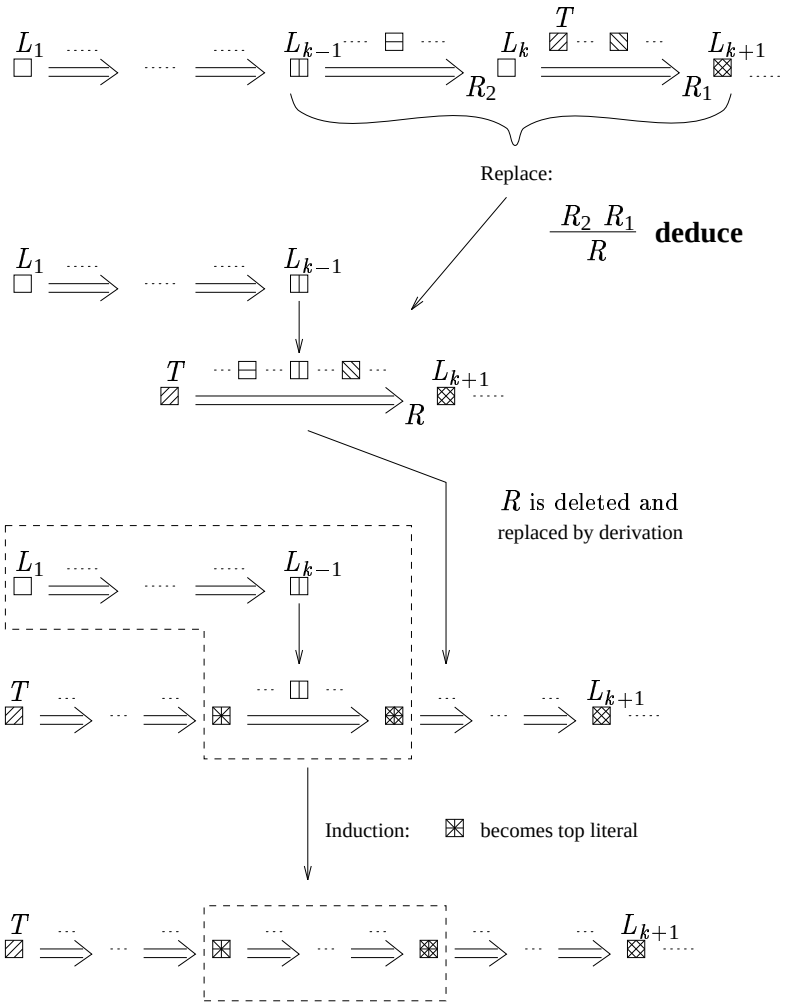

Fig. 17. Illustration of proof of Lemma 38 (ground case).

Together with $L'_k\gamma_1 = L_k = L''_k\gamma_2$ it follows that $L'_k\gamma_1\gamma_2 = L_k = L''_k\gamma_1\gamma_2$. In other words, $\gamma_1\gamma_2$ is a unifier. Hence there exists a MGU σ and a substitution δ such that $\gamma_1\gamma_2 = \sigma\delta|_{dom(\gamma_1\gamma_2)}$. By the existence of this MGU, the **Deduce** transformation rule can be applied to R_1 and R_2 by unifying L'_k and L''_k with σ. The result is the inference rule $R := (T', L'_{k-1}, M'_{k-1}, M'_k \rightarrow L'_{k+1})\sigma$. Since **Deduce** is a mandatory transformation rule and $\mathcal{I}$ is completed (1) $R \in \mathcal{I}$ or (2) R is $\succ_{Lin}$-redundant in $\mathcal{I}$. In case (1) the derivation D_2 in (16) can be replaced by the one step derivation

$$L_{k-1} \overset{T\ M_{k-1}\ M_k}{\Longrightarrow}_{(T', L'_{k-1}, M'_{k-1}, M'_k \rightarrow L'_{k+1})\sigma\delta} L_{k+1}$$

using $R\delta$. Since δ is a ground substitution $R\delta \in \mathcal{I}^g$ and the replacement results in a $\mathcal{I}^g$ derivation with same structure. Furthermore, its top distance is $k-1$. Hence we can apply the induction hypothesis to obtain the desired derivation.

In case (2) things are more complicated. First consider the $(\mathcal{I} \cup \{R\})^g$-derivation

$$T \overset{L_{k-1}\ M_{k-1}\ M_k}{\Longrightarrow}_{R\delta}\ L_{k+1}$$

Since R is $\succ_{Lin}$-redundant in $\mathcal{I}$ by definition there exists a $\mathcal{I}^g$-derivation

$$D_4 = (\ T \Longrightarrow^*_{\mathcal{I}^g,\, M_{k-1} \cup M_k \cup \{L_{k-1}\}}\ L_{k+1})$$

Note that $M_{k-1}, M_k \subseteq M$. Now concatenate to $D_5 := D_4 \cdot D_3$ and obtain a linear derivation of the form

$$D_5 = (\ T \Longrightarrow^*_{\mathcal{I}^g,\, M \cup \{L_{k-1}\}}\ L_n)$$

Next the applications of the literal $\{L_{k-1}\}$ in D_5 have to be eliminated. This can be done one at another by the procedure described below. Once this is done we obtain the desired linear derivation of the form $T \Longrightarrow^*_{\mathcal{I}^g,\, M} L_n$.

If L_{k-1} is not used in D_5 we are done. Otherwise let

$$D_6 = D_5|_{\lambda,l,l+1} = (K_l \overset{L_{k-1}\ N_l}{\Longrightarrow}_{R_l,\gamma_l}\ K_{l+1})$$

be such a derivation step using L_{k-1}. Swapping K_l and L_{k-1} implies the existence of the $\mathcal{I}^g$-derivation

$$D_7 = (L_{k-1} \overset{K_l\ N_l}{\Longrightarrow}_{R_l,\gamma_l}\ K_{l+1})$$

Now concatenate $D_1 \cdot D_7$ to obtain the $\mathcal{I}^g$-derivation

$$D_8 = (L_1 \overset{M_1}{\Longrightarrow} L_2 \ldots \overset{M_{k-2}}{\Longrightarrow} L_{k-1} \overset{K_l\ N_l}{\Longrightarrow}_{R_l,\gamma_l}\ K_{l+1})$$

The top distance of K_l in D_8 is $k-1$ hence we can apply the induction hypothesis to D_8 and obtain a linear $\mathcal{I}^g$-derivation

$$D_9 = (K_l \Longrightarrow^*_{\mathcal{I}^g,\, M}\ K_{l+1})$$

Next build $D_10 := D_5[D_9]_{\lambda,l,l+1}$, i.e. replace the derivation step using L_{k-1} by a derivation not using L_{k-1}. Hence the number of applications of the literal L_{k-1} has decreased by 1, which guarantees the termination of the just described procedure when applied repeatedly in order to eliminate all applications of L_{k-1}. Hence the desired derivation exists.

In order to prove the lifting lemma below we need the following lemma:

LEMMA 47. *Let α, β be substitutions and M be a literal set. Then*

$$\alpha(\beta|_{var(M\alpha)})|_{var(M)} = \alpha\beta|_{var(M)}$$

Proof. Let x be a variable. It suffices to show that both substitutions yield the same result when applied to x. We distinguish two disjoint cases.

1. $x \notin var(M)$. Trivial, since both substitutions yield x.
2. $x \in var(M)$. By definition of restriction of substitution it suffices to show

$$x\alpha(\beta|_{var(M\alpha)}) = x\alpha\beta$$

From $x \in var(M)$ it follows $var(x\alpha) \subseteq var(M\alpha)$ (*). Now we compute

$$x\alpha\beta = x\alpha(\beta|_{var(x\alpha)}) \stackrel{(*)}{=} x\alpha(\beta|_{var(M\alpha)})$$

which was to be shown.

Now we can prove the lifting lemma.

LEMMA 43. **(Lifting lemma for linear refutations)** *Let L_1 be a literal, M be a literal set and γ be a ground substitution for L_1 and M. If there exists a non-trivial linear refutation $L_1\gamma \Longrightarrow^{*}_{\mathcal{I}^g,M\gamma}$ false then there exists a linear first-order refutation $L_1 \Longrightarrow^{\dagger}_{\mathcal{I},M,\sigma}$ false such that $\sigma \leq \gamma \quad [var(M)]$.*

Proof. Let the given refutation be

$$D \;=\; (L_1\gamma \stackrel{M_1\gamma}{\Longrightarrow}_{(L_1',M_1'\rightarrow L_2')\gamma'} L_2 \stackrel{M_2\gamma}{\Longrightarrow} L_3 \ldots L_n \stackrel{M_n\gamma}{\Longrightarrow} false)$$

Induction on the length n of the derivation.

Base case: If $n = 1$ then define $\gamma'' = \gamma\gamma'$. Since the used inference rule is a new variant we can assume that the domains of γ and γ' are disjoint. Hence it holds that $L_1\gamma'' = L_1'\gamma''$ and $M_1\gamma'' = M_1'\gamma''$. Thus γ'' is a unifier for $\{\!|L_1|\!\} \cup M_1$ and $\{\!|L_1'|\!\} \cup M_1'$. Thus there also exists a MGU σ_1 and a substitution δ_1 such that $\sigma_1\delta_1|_{dom(\gamma'')} = \gamma''$ (*). Using this MGU we can build the first-order refutation

$$D' \;=\; (L_1 \stackrel{M_1}{\Longrightarrow}_{L_1',M_1'\rightarrow L_2',\sigma_1} false)$$

with answer substitution σ_1. It remains to show that σ_1 satisfies the claimed property. From (*) and $\gamma'' = \gamma\gamma'$ it follows $\sigma_1\delta_1|_{dom(\gamma)} = \gamma''|_{dom(\gamma)} = \gamma$. Since γ acts on every variable in M it holds $dom(\gamma) \supseteq var(M)$, and thus $\sigma_1\delta_1|_{var(M)} = \gamma|_{var(M)}$ follows. But then by definition $\sigma_1 \leq \gamma \quad [var(M)]$ which was to be shown.

Induction step: For the induction step let $n > 1$ and suppose the result to hold for derivations with length $< n$. Consider the first derivation step in D. Define $\gamma'' = \gamma\gamma'$ in the same way as for the base case. Additionally to the properties for γ'' given above, it holds that $L_2'\gamma'' = L_2 = L_2'\gamma$. Define again σ_1 and δ_1 in the same way and with the same properties as in the base case. Now deleting the first derivation step from D and using $\sigma_1\delta_1$ instead of γ and γ' results in the refutation

$$L_2'\sigma_1\delta_1 \stackrel{M_2\sigma_1\delta_1}{\Longrightarrow} L_3 \ldots L_n \stackrel{M_n\sigma_1\delta_1}{\Longrightarrow} false$$

By the induction hypothesis we can lift this (still linear) refutation to a (linear) refutation

$$L_2'\sigma_1 \overset{M_2\sigma_1}{\Longrightarrow}_{R_2,\sigma_2} L_3 \ldots L_n \overset{M_n\sigma_1\sigma_2\cdots\sigma_{n-1}}{\Longrightarrow}_{R_n,\sigma_n} false$$

with answer substitution

$$\sigma_2\sigma_3\cdots\sigma_n \leq \delta_1 \quad [var(M\sigma_1)]$$

i.e. for some δ, $\sigma_2\sigma_3\cdots\sigma_n\delta|_{var(M\sigma_1)} = \delta_1|_{var(M\sigma_1)}$ (**). Since σ_1 is an appropriate unifier (as in the base case) we can prepend this derivation with the first-order derivation step $L_1 \overset{M_1}{\Longrightarrow}_{L_1', M_1' \to L_2', \sigma_1} L_2'\sigma_1$ to obtain

$$L_1 \overset{M_1}{\Longrightarrow}_{L_1', M_1' \to L_2', \sigma_1} L_2'\sigma_1 \overset{M_2}{\Longrightarrow}_{R_2,\sigma_2} L_3 \ldots L_n \overset{M\sigma_1\sigma_2\cdots\sigma_{n-1}}{\Longrightarrow}_{R_n,\sigma_n} false$$

which is a linear first-order refutation of L_1 as desired with answer substitution $\sigma_1\sigma_2\sigma_3\cdots\sigma_n$. It remains to show that the answer substitution satisfies the claimed property. Now we compute

$$\begin{array}{rcl}
\sigma_1\sigma_2\sigma_3\cdots\sigma_n\delta|_{var(M)} & \overset{\text{Lemma 47}}{=} & \sigma_1(\sigma_2\sigma_3\cdots\sigma_n\delta|_{var(M\sigma_1)})|_{var(M)} \\
& \overset{(**)}{=} & \sigma_1(\delta_1|_{var(M\sigma_1)})|_{var(M)} \\
& \overset{\text{Lemma 47}}{=} & \sigma_1\delta_1|_{var(M)} \\
\text{(see base case)} & = & \gamma|_{var(M)}
\end{array}$$

But then by definition $\sigma_1 \leq \gamma \quad [var(M)]$ which was to be shown.