MAX-PLANCK-INSTITUT FÜR INFORMATIK

Ordered Chaining Calculi for First-Order Theories of Binary Relations

Leo Bachmair
Harald Ganzinger

MPI–I–95–2–009 October 1995

INFORMATIK

Im Stadtwald
D 66123 Saarbrücken
Germany

Authors' Addresses

Leo Bachmair
Department of Computer Science
SUNY at Stony Brook
Stony Brook, NY 11794, U.S.A.
`leo@sbcs.sunysb.edu`

Harald Ganzinger
Max-Planck-Institut für Informatik
Im Stadtwald
D-66123 Saarbrücken, Germany
`hg@mpi-sb.mpg.de`
`http://www.mpi-sb.mpg.de/~hg/`

Publication Notes

This paper has been submitted for publication elsewhere and will be copyrighted, if accepted.

Acknowledgements

This research was supported in part by the German Science Foundation (Deutsche Forschungsgemeinschaft) under grant Ga 261/4-1, by the German Ministry for Research and Technology (Bundesministerium für Forschung und Technologie) under grant ITS 9102/ITS 9103 and by the ESPRIT Basic Research Working Group 6028 (Construction of Computational Logics). The first author was also supported by the Alexander von Humboldt Foundation.

Abstract

We propose inference systems for binary relations with composition laws of the form $S \circ T \subseteq U$ in the context of resolution-type theorem proving. Particulary interesting examples include transitivity, partial orderings, equality and the combination of equality with other transitive relations. Our inference mechanisms are based on standard techniques from term rewriting and represent a refinement of chaining methods. We establish their refutational completeness and also prove their compatibility with the usual simplification techniques used in rewrite-based theorem provers. A key to the practicality of chaining techniques is the extent to which so-called variable chainings can be restricted. We demonstrate that rewrite techniques considerably restrict variable chaining, though we also show that they cannot be completely avoided in general. If a binary relation under consideration satisfies additional properties, such as symmetry, further restrictions are possible. In particular, we discuss orderings and partial congruence relations.

Keywords

Automated Theorem Proving, First-Order Logic, Transitive Relations, Chaining, Orderings, Rewrite Techniques, Simplification

Contents

1 Introduction

Rewrite techniques, as exemplified by completion procedures, have been successfully applied to many problems in equational theorem proving. The theoretical foundation of completion is the use of rewrite rules, guided by well-founded syntactic orderings, in such a way that certain commutation properties are satisfied. Building ordered rewriting into an inference system, through the concept of rewrite proofs, forward reasoning (search) is effectively restricted and combined with don't-care nondeterministic backward reasoning (simplification). The explicit generation of the congruence closure of a set of equations is usually avoided. This is one main advantage of rewrite techniques over methods based on resolution alone. From a practical point of view the simplification techniques afforded by rewriting are even more important. In the context of automated theorem proving, rewrite techniques have been typically applied to equivalence relations. In this paper we investigate their application to arbitrary (e.g., non-symmetric) transitive relations and design corresponding inference systems for first-order theories with transitive relations, using resolution as the underlying inference mechanism.

Resolution, in its different variants, forms the core of many current automated reasoning systems. (For a recent exposition of the theory of resolution see Bachmair and Ganzinger 1995b.) By and large such resolution refinements as hyper-resolution, ordered resolution, or the set-of-support strategy are quite useful in practice, but for logical theories with transitive relations, such as logics with equality or inequality relations, they are not very effective. Special techniques have been devised for such theories: chaining and variable elimination for inequalities, and paramodulation, which can be thought of as a form of subterm chaining, for equality. Chaining essentially encodes certain resolution steps with the transitivity axiom. For example, if S is a transitive relation, chaining allows one to derive $u\sigma \, S \, v\sigma$ from $u \, S \, s$ and $t \, S \, v$, where σ is a most general unifier of s and t. The completeness of this inference mechanism follows immediately from the completeness of resolution with selection (Bachmair and Ganzinger 1994b).

We propose the following refinement of chaining: first compare the term $s\sigma$ (which is identical to $t\sigma$) to both $u\sigma$ and $v\sigma$ (using a given well-founded ordering on terms) and perform the chaining inference only if $s\sigma$ is maximal. This form of "ordered chaining" can be viewed as an application of standard rewrite techniques. More precisely, we argue that it corresponds to a completion process, aimed at deriving enough rewrite rules so that the given transitive relation can be described by corresponding rewrite proofs. The completeness of ordered chaining is based on a suitable commutation property of the rewrite relations involved which is achieved through completion. In formalizing this approach we use standard methods from the theory of term rewriting. However, we wish to apply this approach not only to unit clauses, as is the case with standard completion procedures, but to general clauses, and therefore have to integrate the notion of rewriting in this more general context. This is relatively straightforward for Horn clauses, but more difficult in the case of disjunctions of positive literals, due to the additional degree of non-determinism in the rewrite relations and certain subtle dependencies between them. We will discuss the two cases separately.

The practicality of chaining methods crucially depends on the extent to which a particularly prolific form of chaining, called variable chaining (where s or t is a variable), can be restricted. Ordered chaining considerably cuts down the number of possible variable chainings, as the required maximality condition for terms can only be satisfied if the variable is unshielded, i.e., occurs only as an argument of the predicate S, but not as an argument of a function symbol or any other predicate symbol. However, we also show that some variable chainings are needed for transitive relations in general, though further restriction are possible if additional properties such as symmetry are satisfied. For instance, in the case of equivalence relations we obtain calculi in which chaining "into" a variable is not needed.

Transitivity is a particular instance of a composition law for binary relations of the form $S \circ T \subseteq U$. The chaining techniques in this paper will be applicable to a large class of such composition laws. In order to introduce the main ideas in a way that does not require much technical detail, we will first, in Section 3, describe our methods for the particularly simple case of Horn clauses over one transitive relation. In Section 4 the methods will be extended to composition laws in general and to full first-order clauses. Refutational completeness is established for an appropriately extended inference system and in the presence of a notion of redundancy that covers the usual simplification techniques used in rewrite-based theorem provers. Redundancy is the central concept when tailoring the general system to particulary interesting cases of binary relations. Each case will be investigated with the goal to further optimize the inference system and to avoid variable chaining as much as possible. In particular we consider (partial) equivalence and congruence relations in Section 5.1 and total orderings in Section 6; and conclude with a summary and suggestions for further research.

This paper combines results that we have annonced in (Bachmair and Ganzinger 1994a) and (Bachmair and Ganzinger 1994c) into a single framework with a uniform and complete presentation.

2 Preliminaries

We consider first-order languages with function symbols, predicate symbols, and variables. A *term* is an expression $f(t_1, \ldots, t_n)$ or x, where f is a function symbol of arity n, x is a variable, and $t_1, \ldots, t_n$ are terms. An *atomic formula* (or simply *atom*) is an expression $P(t_1, \ldots, t_n)$, where P is a predicate symbol of arity n and $t_1, \ldots, t_n$ are terms. A *literal* is an expression A (a *positive* literal) or $\neg A$ (a *negative* literal), where A is an atomic formula. Atoms from binary predicate symbols S are often written in infix notation as $u\,S\,v$. A *clause* is a finite multiset of literals. We write a clause by listing its literals, $\neg A_1, \ldots, \neg A_m, B_1, \ldots, B_n$; or as a disjunction $\neg A_1 \vee \cdots \vee \neg A_m \vee B_1 \cdots \vee B_n$; or as a sequent $A_1, \ldots, A_m \rightarrow B_1, \ldots, B_n$. An expression is said to be *ground* if it contains no variables.

By a (*Herbrand*) *interpretation* we mean a set I of ground atomic formulas. We say that an atom A is *true* (and $\neg A$, *false*) *in* I if $A \in I$; and that A is *false* (and $\neg A$, *true*) in I if $A \notin I$. A ground clause is *true* in an interpretation I if at least one of its literals is true in I; and is *false* otherwise. In general, a clause is said to be true in I if all its ground instances are true. The *empty clause* is false in every interpretation. We say that I is a *model* of a set of clauses N (or that N is *satisfied* by I) if all elements of N are true in I. Occasionally, a model of N will also be called an N-*interpretation*. A set N is *satisfiable* if it has a model, and *unsatisfiable* otherwise. For instance, any set containing the empty clause is unsatisfiable.

3 Transitive Relations

In this section we will consider the satisfiability problem for sets of Horn clauses in languages with transitive relations, that is, predicates S that satisfy

$$x\,S\,y\,,\; y\,S\,z \rightarrow x\,S\,z.$$

Such clauses present difficulties for resolution-based provers. A transitivity axiom can be resolved with itself to yield a new clause

$$x\,S\,y_1\,,\; y_1\,S\,y_2\,,\; y_2\,S\,z \rightarrow x\,S\,z$$

that can again be resolved with the original clause, to yield

$$x\,S\,y_1\,,\; y_1\,S\,y_2\,,\; y_2\,S\,y_3\,,\; y_3\,S\,z \rightarrow x\,S\,z$$

and so on. Certain refinements of resolution provide better control in this regard. For instance, the self-resolution that is problematic with transitivity, is automatically excluded by resolution with selection (Bachmair and Ganzinger 1994b): if we select the two negative literals in the transitivity axiom, the only kind of inference in which it can participate are inferences of the form

$$\frac{C\,,\; u\,S\,s \quad D\,,\; t\,S\,v \quad \neg(x\,S\,y)\,,\; \neg(y\,S\,z)\,,\; x\,S\,z}{C\sigma\,,\; D\sigma\,,\; u\sigma\,S\,v\sigma}$$

where σ is a most general unifier of s and t. The key here is that it is never necessary to resolve with the non-selected (i.e., positive) literal of the transitivity axiom (for details, see Bachmair and Ganzinger 1995b). Therefore we may encode transitivity as an inference rule:

Chaining:

$$\frac{C\,,\; u\,S\,s \quad D\,,\; t\,S\,v}{C\sigma\,,\; D\sigma\,,\; u\sigma\,S\,v\sigma}$$

where σ is a most general unifier of s and t.

Unfortunately, the chaining rule, which was already proposed by Slagle (1972), is not completely satisfactory, as it explicitly generates the transitive closure of the binary relation described by S. For example, if $a\,S\,b$ and $b\,S\,c$ and $c\,S\,d$ are given, chaining will generate the additional unit clauses $a\,S\,c$ and $b\,S\,d$ and $a\,S\,d$.

We will next outline how standard rewrite techniques can be used in this context to more effectively restrict the chaining rule. Rewrite systems provide for a more compact representation of transitive (closures of) relations and are one of the more efficient techniques for reasoning about chains of terms, such as $a\,S\,b\,S\,c\,S\,d$ in the above example. Rewrite techniques have typically been applied to equivalence and congruence relations. We also discuss non-symmetric relations, developing an approach similar in spirit to what has been called bi-rewriting by Levy and Agustí (1993).

3.1 Commutation

If S is a binary relation, we denote by S^{-1} its inverse, by S^{+} its transitive closure, and by S^{*} its reflexive-transitive closure. (A binary relation S is called *reflexive* if xSx for all x; *irreflexive* if xSx for no x; and *non-reflexive* if it is not reflexive.) The *composition* of two binary relations S and T is denoted by $S \circ T$. (x, y) is in $S \circ T$ if and only if there exists a z such that (x, z) is in S and (z, y) is in T. Our objective is to represent transitive (closures of) relations by means of well-founded rewrite relations.[1] Two relations S_1 and S_2 are employed for that purpose.

We say that S_1 *commutes* with S_2 if $S_2 \circ S_1 \subseteq (S_1^+ \circ S_2^*) \cup S_2^+$.[2]

Proposition 1 *If $S_1 \cup S_2^{-1}$ is a well-founded binary relation and S_1 commutes with S_2, then the two relations $(S_1 \cup S_2)^+$ and $(S_1^+ \circ S_2^*) \cup S_2^+$ are identical.*

The proposition can be proved with standard methods from rewriting theory.[3] It provides the starting point for our investigations.

Let $\succ$ be a (strict) ordering and $\succeq$ be its reflexive closure. We denote by $\Rightarrow_S$ the intersection of S with $\succ$; by $\Leftarrow_S$ the intersection of S with the inverse $\prec$ of the relation $\succ$; and by $\Leftrightarrow_S$ the reflexive part of S, that is, the intersection of S with the identity relation. If the ordering $\succ$ is total, then $S = \Leftrightarrow_S \cup \Rightarrow_S \cup \Leftarrow_S$. We drop the subscript when the relation is known from the context. Slightly adapting Proposition 1, we obtain:

Proposition 2 *Let S be a binary relation and $\succ$ be a well-founded ordering that is total on the given domain. If the relation $\Rightarrow_S \cup \Leftrightarrow_S$ commutes with $\Leftarrow_S \cup \Leftrightarrow_S$ then $S^+ = \Leftrightarrow_S \cup (\Rightarrow_S^+ \circ \Leftarrow_S^*) \cup \Leftarrow_S^+$.*

This result shows that under suitable commutation properties the non-reflexive part of a transitive relation can be described by a certain kind of rewrite proofs with $\Rightarrow$ and $\Leftarrow$.

In the logic context a relation is often represented by ground atoms in a Herbrand interpretation I. To simplify matters technically, for the remainder of this section we shall assume that S is the only predicate symbol, and that it denotes a transitive relation. If I satisfies the transitivity law T_S for S then I is called a *transitivity interpretation*. If I is a Herbrand interpretation, that is, a set of ground atoms (for S), and if $\succ$ is a wellfounded ordering on ground atoms, by analogy with the notation used before, we shall write $\Rightarrow_I$, $\Leftarrow_I$, and $\Leftrightarrow_I$, respectively, to denote the intersection of S (as defined by I) with the relations $\succ$, $\prec$, and $=$. A *rewrite proof* of $u\,S\,v$ in I is either of the form

$$u \Leftrightarrow_I u,$$

provided $u = v$, or else is a non-empty sequence of rewrites

$$u = u_0 \Rightarrow_I \cdots \Rightarrow_I u_m = v_n \Leftarrow_I \cdots \Leftarrow_I v_0 = v,$$

where $m + n \geq 1$. We write $u \Downarrow_I v$ when there exists a rewrite proof for $u\,S\,v$ in I. A sequence $u \Leftarrow_I t \Rightarrow_I v$, on the other hand, is called a *peak*. The peak is said to *commute* if $u \Downarrow_I v$. By a *plateau* we mean a sequence of rewrites $t \Leftrightarrow_I t \Rightarrow_I v$ or $u \Leftarrow_I t \Leftrightarrow_I t$. Any plateau can be transformed into a rewrite proof by simply deleting the $\Leftrightarrow$-step. A sequence of rewrites is a rewrite proof if and only if it contains no peak and no plateau as a subsequence.

By the *rewrite closure* of I we mean the set $\{u\,S\,v : u \Downarrow_I v\}$. As a consequence of Proposition 2 we obtain:

[1] The relations we talk about are abstract reduction systems in the sense of Klop (1992), but not necessarily rewrite relations in the sense of Dershowitz and Jouannaud (1990), in that they need not be closed under context application.

[2] The relation $(S_1^+ \circ S_2^*) \cup S_2^+$ is identical to $(S_1^+ \circ S_2^*) \cup (S_1^* \circ S_2^+)$.

[3] Commutation properties, with a view to their application to termination problems, have been discussed in Bachmair and Dershowitz 1986, Bellegarde and Lescanne 1987, Geser 1991.

Proposition 3 *Let $\succ$ be well-founded and total on ground terms. The rewrite closure of I is a transitivity interpretation if and only if all peaks in I commute.*

For example, if I contains the atoms $a\,S\,b$ and $b\,S\,c$ and $c\,S\,d$, and if $a \succ b \succ c \succ d$, then $a \Rightarrow_I b$ and $b \Rightarrow_I c$ and $c \Rightarrow_I d$. The relation $\Rightarrow_I$ evidently commutes with the (empty) relation $\Leftarrow_I$. If we choose the ordering $b \succ c \succ d \succ a$, the rewrite steps are $a \Leftarrow_I b$ and $b \Rightarrow_I c$ and $c \Rightarrow_I d$, and there is a peak, $a \Leftarrow_I b \Rightarrow_I c$, that does not commute. If, through *ordered chaining*, we generate $a\,S\,c$, the corresponding rewrite step $a \Leftarrow c$ makes the peak converge. Chaining with suitable ordering restrictions provides the basis for a completion process of which equational completion is a special case in which the two rewrite relations $\Rightarrow$ and $\Leftarrow^{-1}$ are the same, cf., Section 5.1. Completion always succeeds for finite (ground) I. Equational completion procedures, such as ordered completion or conditional completion, have been described by inference rules, e.g., Bachmair, Dershowitz and Plaisted 1989, Ganzinger 1991, Dershowitz 1991. We will devise similar (refutationally complete) inference systems for transitive relations in general.

One important difference between a convergent rewrite system in the equational case and commutation of one rewrite system over another in the case of a non-symmetric relation S exists, though. A convergent system admits unique normal forms. Searching for a rewrite proof for a given pair of ground terms is don't-care nondeterministic in that no backtracking is required. Commutation is weaker in this regard. In general one may have more than one "minimal" rewrite proof for any two given terms, and there may be no deterministic strategy of finding anyone of them.

3.2 Negative chaining

Having computed sufficiently many consequences by ordered chaining, for negative literals in clauses we need to enumerate their rewrite proofs. Take the above example and add another clause $\neg(a\,S\,d) \vee d\,S\,e$. By transitivity we have $a\,S\,d$, which we may resolve with the additional clause to obtain $d\,S\,e$. However, with ordered chaining, we do not derive $a\,S\,d$ directly, but only obtain enough inequalities for a suitable rewrite proof of $a\,S\,d$. We need an inference mechanism which solves negative inequalities and, as a side effect, generates enough inequalities for a rewrite proof of $d\,S\,e$. In the equational case, negative literals in clauses are solved by narrowing-like inferences which enumerate equational rewrite proofs. In the general case of a transitive relation, the corresponding inference mechanism will be called *negative chaining*.

3.2.1 Admissible Clause Orderings

We combine the chaining rules into an ordered resolution calculus, and for that purpose have to require an ordering on terms, literals and clauses. A well-founded and total ordering $\succ$ on ground terms and ground literals is called *admissible* if it is compatible with a certain *complexity measure* c_L on ground literals L. By *compatibility* we mean that whenever $c_L \succ c_{L'}$ then $L \succ L'$, for any two ground literals L and L'. Throughout this paper we will use several such complexity measures, any of which will be a refinement of some previously defined measure. Results which are obtained for admissible orderings then also apply to orderings that are compatible with the refined measure.

For the purpose of this section, the following complexity measure will suffice: Suppose L is of the form $u\,S\,v$ or $\neg(u\,S\,v)$. Then $c_L = (max_L, p_L, s_L)$, where (i) max_L

denotes the maximal term of u and v with respect to $\succ$; (ii) p_L is 1, if L is negative, and 0 otherwise; and (iii) s_L is 1 if $u \succeq v$, and 0 otherwise. We assume that the lexicographic combination of $\succ$ on terms with $1 > 0$ is used to compare the complexity triples. For example, if $s \succ t$, then the complexity of $s\,S\,t$ is $(s, 0, 1)$, whereas the complexity of $\neg(t\,S\,s)$ is $(s, 1, 0)$. The maximal term of a literal is the main criterion for any admissible literal ordering. Rewriting the maximal term into a smaller term will always yield a smaller literal. We observe that a negative literal is considered more complex than a positive literal with the same maximal term. Moreover we have stipulated that the argument terms of an atom be lexicographically compared.

An ordering on ground clauses is called *admissible* if it is the multiset extension of an admissible ordering $\succ$ on literals. We will also use $\succ$ to denote the clause ordering. We assume that orderings on ground expressions are extended to non-ground expressions as follows: $E \succ E'$ if and only if $E\sigma \succ E'\sigma$, for all ground instances $E\sigma$ and $E'\sigma$. Thus, we have $E \not\preceq E'$ if $E'\sigma \succ E\sigma$, for some ground instances $E\sigma$ and $E'\sigma$.[4] We say that a literal L is *maximal with respect to a clause* C if $L' \succ L$, for no literal L' in C; and that L is *strictly maximal* with respect to C if $L' \succeq L$, for no L' in C. Given an admissible ordering on terms and literals, its extension to an admissible ordering on clauses is uniquely defined by the above stipulations.

A great variety of admissible literal and clause orderings exists for arbitrary signatures of function symbols.[5] Selecting an appropriate such ordering is one of the most important ways in which one can take an influence on the search strategy of a prover that is based on rewriting methods. Defining selection strategies for negative literals is the second major high-level primitive for guiding search. In this paper we will not consider selection. It is not hard to see that the results about selection in the equational case (Bachmair and Ganzinger 1994b) can be extended to the framework that we consider in this paper. The calculi that we present here will not include selection so as to not further complicate matters technically.

3.2.2 Inferences of Ordered Chaining

We are now prepared for defining the inference rules of a chaining calculus.[6] We assume $\succ$ to denote some admissible ordering on terms, literals and clauses.

Ordered Chaining:

$$\frac{C, u\,S\,s \quad D, t\,S\,v}{C\sigma, D\sigma, u\sigma\,S\,v\sigma}$$

where (i) σ is the most general unifier of s and t, (ii) $u\sigma\,S\,s\sigma$ is strictly maximal with respect to $C\sigma$, (iii) $t\sigma\,S\,v\sigma$ is strictly maximal with respect to $D\sigma$, (iv) $u\sigma \not\preceq s\sigma$, and (v) $v\sigma \not\preceq t\sigma$.

Negative Chaining:

$$\frac{C, \neg(u\,S\,s) \quad D, t\,S\,v}{C\sigma, D\sigma, \neg(v\sigma\,S\,\sigma)}$$

[4] Depending on the ordering, this extension to non-ground expressions may or may not be decidable. In the latter case one will have to employ a safe and decidable approximation.

[5] For a survey on orderings see Dershowitz (1987).

[6] As usual we implicitly assume that the premises of an inference have no common variables. If necessary, the variables in one premise are renamed. Thus, it is also possible to use different variants of a clause as premises in one inference.

where (i) σ is the most general unifier of u and t, (ii) $\neg(u\sigma\, S\, s\sigma)$ is maximal with respect to $C\sigma$, (iii) $t\sigma\, S\, v\sigma$ is strictly maximal with respect to $D\sigma$, (iv) $v\sigma \not\succeq t\sigma$, (v) $s\sigma \not\succeq u\sigma$, and (vi) $s\sigma \neq v\sigma$; and

$$\frac{C\,,\, \neg(u\, S\, s) \quad D\,,\, t\, S\, v}{C\sigma\,,\, D\sigma\,,\, \neg(u\sigma\, S\, t\sigma)}$$

where (i) σ is the most general unifier of s and v, (ii) $\neg(u\sigma\, S\, s\sigma)$ is maximal with respect to $C\sigma$, (iii) $t\sigma\, S\, v\sigma$ is strictly maximal with respect to $D\sigma$, (iv) $u\sigma \not\succ s\sigma$,[7] (v) $t\sigma \not\succeq v\sigma$, and (vi) $u\sigma \neq t\sigma$.

The resolution rule takes care of one-step rewrite proofs for negative literals:

Ordered Resolution:

$$\frac{C\,,\, A \quad D\,,\, \neg B}{C\sigma\,,\, D\sigma}$$

where (i) σ is the most general unifier of A and B, (ii) $A\sigma$ is strictly maximal with respect to $C\sigma$, and (iii) $B\sigma$ is maximal in $D\sigma$.

The calculus of all these inference rules is denoted by B_S, or simply B (the Basic chaining calculus).

3.3 Refutational Completeness for Horn Clauses

Clauses with at most one positive literal are called *Horn clauses.* The calculi B_S are refutationally complete for such clauses. For the completeness proof we adapt the model construction approach of Bachmair and Ganzinger (1990) (see also Bachmair and Ganzinger 1994b).

Given a set N of ground clauses, we define a corresponding Herbrand interpretation I using induction on $\succ$. More precisely, we define for each clause C an interpretation I_C, intended to be a model for clauses smaller than C, and a set E_C that is designed to turn I_C into a model of C as well. Since ordered chaining is supposed to achieve that atoms can be proved by rewrite proofs, the partial interpretations I_C, as well as the final I, will be represented as the rewrite closures of suitable sets R_C and R.

Formally, for every clause C in N we define R_C to be the set $\bigcup_{C \succ D} E_D$. By I_C we denote the rewrite closure of R_C. Furthermore, if C is a clause $C' \vee A$, where A is a positive literal and $A \succ C'$,[8] and C is false in I_C then $E_C = \{A\}$. In that case, we also say that C is *productive* and that it *produces* A. In all other cases, $E_C = \emptyset$. Finally, let R be $\bigcup_C E_C$ and let I be the rewrite closure of R.

In what follows we shall also use the notation R^C for $R_C \cup E_C$ and I^C for the rewrite closure of R^C. Hence I_C is the partial interpretation defined by clauses smaller that C, whereas I^C additionally includes the effect of C in this construction. All these interpretations are rewrite closures of suitable rule sets so that an atom is true if and only if it has a rewrite proof in the corresponding rule set.

The following lemma expresses an important monotonicity property of our construction:

Lemma 1 *If a ground clause C (which need not be in N) is true in some interpretation I_D or I^D, where $D \succeq C$, then C is also true in I and in any interpretation $I_{D'}$ and $I^{D'}$ with $D' \succ D$. Furthermore, if a clause C is productive, then it is true in I^C.*

[7] The case $u\sigma = s\sigma$ needs to be dealt with by only one of the forms of negative chaining.

[8] In writing $A \succ C'$ we view A to denote a positive unit clause.

Proof. The proof is straightforward and based on the following observations: Atoms which are true in some partial interpretation I_D or I^D remain true in any "later" interpretation $I_{D'}$ or $I^{D'}$ with $D' \succ D$, as the rule sets of which these rewrite closures are formed increase monotonically. Compatible orderings $\succ$ on literals are designed so that whenever a negative literal $\neg A$ occurs in C and A is false in I_D or I^D with $D \in N$ and $D \succeq C$, then A remains false in I. If such an atom A is false in I_D or I^D then A has no rewrite proof in rules generated "up to D". If $D \succeq C$, rules that are generated into later interpretations $I_{D'}$ have too large "left-hand sides" to be able to reduce A. $\Box$

This lemma will often be applied in its contrapositive form to infer that C is false in I_C and I^C whenever it is false in $I_{D'}$ or $I^{D'}$, for some $D' \succ C$.

The next lemma states a continuity property of the model construction.

Lemma 2 *Let D be a clause in N and let C be a ground clause (which need not be in N). If C is true in all interpretations $I^{D'}$ with $D \succ D'$, then C is also true in I_D.*

Proof. If a ground atom A is true in I_C then has a rewrite proof over R_C, hence is true in $I^{D'}$ where $D' \prec C$ is the clause that produces the largest atom that occurs in the proof of A. Therefore, if the antecedent of C is true in I_C, it is already true in some such $I^{D'}$. In this case the succedent of C is true in $I^{D'}$ and in I_C. $\Box$

The model construction given above is a rather general concept. It is parametrized by the notion of a rewrite closure (and hence of a rewrite proof). Although in this section we consider Horn clauses only, the construction is independent of that property. Later on we shall refine our inference systems and extend them to full clauses. In these contexts, the respective notions of a rewrite proof will be correspondingly refined, but the model construction per se will remain the same. It is important to note that the two basic properties of the construction as stated in the Lemmas 1 and 2 will hold for any of the notions of rewriting that we consider in this paper.

We say that a set of (possibly non-ground) clauses N is *saturated* (with respect to an inference system J) if it contains all conclusions of inferences (in J) from N.

Lemma 3 *Let N be a set of ground Horn clauses that is saturated with respect to* B *and does not contain the empty clause, and let I be the interpretation constructed from N. Then for every clause C in N we have:*

(1) If C is a productive clause $C' \vee A$, with $A \succ C'$, then C' is false in I.

(2) I^C as well as I_C are transitivity interpretations.

(3) C is true in I^C.

Proof. We use induction on $\succ$. Let C be a ground clause in N, such that (1)–(3) are satisfied for all smaller clauses in N.

(1) Suppose C is a productive clause of the form $C' \vee A$, with $A \succ C'$. Since C is a Horn clause, C' contains only negative literals. Furthermore, C is false in I_C. Thus, if $\neg B$ is a literal in C', then $B \in I_C \subseteq I$, from which we conclude that C' is false in I.

(2) According to Proposition 3 we have to prove that all peaks $u \Leftarrow v \Rightarrow t$ in I^C commute. By the induction hypothesis, peaks from any two rules in R_C commute, hence I_C is a transitivity interpretation. The assertion for I^C is thus trivially true if E_C is empty, or if $E_C = \{t \, S \, v\}$ where $v \succeq t$. (If $v \succ t$, the bigger term occurs on the right side of the inequation. Clauses producing atoms of the form $v \, S \, u$ in which the maximal term occurs on the left side cannot be smaller than C with respect to $\succ$.) Suppose

$C = C' \vee v\,S\,t$ produces $v\,S\,t$ with $v \succ t$. If there is another clause $D = D' \vee u\,S\,v$ that produces uSv with $v \succ u$, then there exists a peak $u \Leftarrow v \Rightarrow t$ in I^C. Now consider ordered chaining by which the clause $C'' = C' \vee D' \vee u\,S\,t$ can be obtained from C and D. Since N is saturated and $C \succ C''$ we may apply the induction hypothesis for (3), to infer that C'' is true in $I^{C''}$. But $C' \vee D'$ is false in I and $R^{C''} \subseteq R_C$, hence $u \Downarrow_{R_C} t$. This establishes the required commutation property.

(3) If C is false in I^C, then it cannot be productive, hence must violate the condition imposed on productive clauses, that is, the maximal literal in C is negative. Let us therefore assume C can be written as $C' \vee \neg A$, with a maximal atom $\neg A$ that is false in I_C. Thus the atom A is true in I_C. If A is produced by some (smaller) clause $D = D' \vee A$, then the resolvent $C'' = D' \vee C'$ of C and D is smaller than C (since $\neg A \succ D'$), contained in N, and, as a consequence of (i), false in I_C and hence false in $I^{C''}$—which contradicts the induction hypothesis. The only remaining possibility is that A is an atom $u\,S\,v$ with $u \Downarrow_{R_C} v$. In that case there exists a suitable productive clause $D \vee u\,S\,v'$ (where $v' \Downarrow_{R_C} v$) or $D \vee u'\,S\,v$ (where $u \Downarrow_{R_C} u'$) which, through negative chaining with C, produces a clause C'' smaller than C that is contained in N and is false in $I^{C''}$, which is again a contradiction. Thus, C must be true in I^C, which completes the proof. □

The lemma indicates that all interpretation I_C and I are transitivity interpretations. As an immediate corollary of the lemma we obtain:

Theorem 1 *If a set of Horn clauses N is saturated with respect to B_S, then the set $N \cup T_S$ is unsatisfiable if and only if it contains the empty clause.*

Proof. If N contains the empty clause, $N \cup T_S$ is unsatisfiable. Suppose N does not contain the empty clause. Let I be the Herbrand interpretation constructed from the set of all ground instances of N. It follows from standard results in resolution theory (usually called "lifting lemmas") that if N is saturated with respect to B, then the set of its ground instances is also saturated. Therefore we may use the above lemma to infer that the relation $\Downarrow_R$ is transitive and that I is a model of $N \cup T_S$. □

We can use this theorem to check the unsatisfiability of sets of Horn clauses over languages with transitive relations: saturate the given set of clauses with respect to the inference system B by applying the inferences in B exhaustively; if the set if unsatisfiable, a contradiction in the form of the empty clause will eventually be generated. In the following section we generalize the ordered chaining calculus B to non-Horn clauses and to forms of composition laws for binary relations which are more general than transitivity. Before, though, we take a first look at the problem of variable chaining.

3.4 Variable chaining

In chaining we combine two inequalities (one of which may be negative) by unifying the first argument of one with the second argument of the other. If one of the two terms thus unified is a variable we speak of a *variable chaining*. More specifically, we speak of chaining *into* a variable if the inequality with the variable is not smaller (with respect to $\succ$) than the other inequality; and speak of chaining *from* a variable if the inequality with the variable is not bigger than the other inequality. Since the unification of a term with a variable not occurring in it always succeeds, variable chaining can be particularly prolific. Fortunately, the combination of certain redundancy criteria (to be defined and proved compatible with chaining below) and ordering constraints drastically cuts down on variable chaining.

Consider, for instance, an inference

$$\frac{C\,,\, u\,S\,x \quad D\,,\, t\,S\,v}{C\sigma\,,\, D\,,\, u\sigma\,S\,v}$$

where $x\sigma = t$. The constraints imposed on an ordered chaining inference require x to be a maximal term in the first premise, which implies that x does not occur in a negative literal and also is *unshielded* in the clause in the sense that it occurs only as an argument of S but not as an argument of any other predicate symbol or any function symbol. Moreover, the inference is redundant if the variable x is *linear* in the first premise, i.e., does not occur in C or u. For in that case the conclusion is $C \vee D \vee u\,S\,v$, and is either properly subsumed by $C \vee u\,S\,x$ or else is identical to $C \vee u\,S\,x$ up to renaming of variables. Similar arguments apply to inferences

$$\frac{C\,,\, u\,S\,s \quad D\,,\, x\,S\,v}{C\,,\, D\sigma\,,\, u\,S\,v\sigma}$$

where the variable occurs as the first argument of an S atom.

In sum, ordered chaining through a variable is only necessary if the variable is non-linear, unshielded, and occurs only in positive literals. For Horn clauses these conditions cannot be satisfied simultaneously, so that ordered chaining through variables is not needed at all. But negative chaining into non-linear unshielded variables is still necessary, even for Horn clauses, as the following example shows. Take the set of clauses

$$\begin{array}{rcl} & \rightarrow & a\,S\,a' \\ & \rightarrow & a'\,S\,c \\ & \rightarrow & b\,S\,b' \\ & \rightarrow & b'\,S\,c \\ a\,S\,x\,,\, b\,S\,x & \rightarrow & \end{array}$$

where $c \succ b' \succ b \succ a' \succ a$. This set of clauses is unsatisfiable. From the first two clauses we obtain $a\,S\,c$; from the third and fourth clause $b\,S\,c$. The two inequalities contradict the last clause. No ordered chaining is possible. We may resolve the third with the last clause, to get $\neg(a\,S\,b')$, from which we get $\neg(a\,S\,b)$ by chaining with the third clause again. Any other inferences require chaining through a variable. In other words, no contradiction can be derived if we exclude variable chaining. If we use the fourth and fifth clause to chain through a variable, we get $\neg(a\,S\,c) \vee \neg(b\,S\,b')$, from which a contradiction can be derived by one more chaining and two resolution steps.

In sections 5.4 and 6.6 we show that chaining into variables can be avoided if the given transitive relation is a partial congruence, or else if it is a dense, total ordering without endpoints.

4 General Binary Relations

We now generalize the results of the previous section to (i) full clauses, (ii) a more general form of composition laws for binary relations (to be defined in Section 4.1), and

(iii) to a suitable concept of redundancy which will be defined in Section 4.2 and that will be compatible with chaining. The completeness results will be valid for a large class of variants of inference systems, differing in how they exploit additional logical properties of relations by specialized derived inferences that make other, less specialized inferences redundant.

4.1 Composition Laws

Transitivity is a special case of a composition law for binary relations. For a generalized chaining we assume a set Co of *composition laws* (or *composition axioms*) of the form

$$x\,S\,y\,,\; y\,T\,z \rightarrow x\,U\,z$$

where S, T, and U are binary relations. We will also write $S \circ T \subseteq U$ in this case. We assume that composition of relations as defined by the laws in Co is *associative*, that is, whenever we have two laws $S \circ T \subseteq U$ and $U \circ V \subseteq W$, then we must also have laws of the form $T \circ V \subseteq V'$ and $S \circ V' \subseteq W$, and vice versa. In that case the consequences of Co, they have the form $S_1 \circ S_2 \ldots \circ S_n \subseteq S$, do not depend on the order in which the basic laws are applied. To simplify the technical treatment we require in addition that in each law $S \circ T \subseteq U$ either $U = S$ or $U = T$. A relation symbol that occurs in a composition law is called a *special* (binary) relation.

Example. The clauses

$$\begin{array}{rcl}
x < y\,,\; y < z & \rightarrow & x < z \\
x \approx y\,,\; y \approx z & \rightarrow & x \approx z \\
x \leq y\,,\; y \leq z & \rightarrow & x \leq z \\
x \approx y\,,\; y < z & \rightarrow & x < z \\
x < y\,,\; y \approx z & \rightarrow & x < z \\
x \approx y\,,\; y \leq z & \rightarrow & x \leq z \\
x \leq y\,,\; y \approx z & \rightarrow & x \leq z \\
x < y\,,\; y \leq z & \rightarrow & x < z \\
x \leq y\,,\; y < z & \rightarrow & x < z
\end{array}$$

define the composition laws for a strict and non-strict ordering $<$ and $\leq$, respectively, together with equality $\approx$. $\square$

We assume again that $\succ$ denotes a given well-founded and total ordering on ground terms. Let I be a set of ground atoms and, for any binary relation S, let $\Rightarrow_I^S$, and $\Leftarrow_I^S$, and $\Leftrightarrow_I^S$, respectively, denote the intersection of the binary relation S as represented by the set of S-atoms in I with $\succ$, $\prec$, and $=$. Generalizing the notions of Section 3.2, a *proof* of $u\,S\,v$ in I is a sequence of the form

$$u = u_0 S_1 u_1\, S_2\, u_2 \ldots u_{n-1}\, S_n\, u_n = v$$

where $n \geq 1$, the $u_i S_i u_{I+1}$ are in I, and $S_1 \circ S_2 \ldots \circ S_n \subseteq S$ follows from Co. The proof is called a rewrite proof if it is of the form

$$u = u_0 \Rightarrow_I^{S_1} u_1 \;\ldots\; u_{k-1} \Rightarrow_I^{S_k} u_k \Leftarrow_I^{S_{k+1}} u_{k+1} \;\ldots\; u_{n-1} \Leftarrow_I^{S_n} u_n = v$$

or of the form

$$u = u_0 \Rightarrow_I^{S_1} u_1 \;\ldots\; u_{k-2} \Rightarrow_I^{S_{k-1}} u_{k-1} \Leftrightarrow_I^{S_k} u_k \Leftarrow_I^{S_{k+1}} u_{k+1} \;\ldots\; u_{n-1} \Leftarrow_I^{S_n} u_n = v$$

with $1 \leq k \leq n$. The existence of a rewrite proof for $u\,S\,v$ in I will be indicated by the notation $u \Downarrow_I^S v$. The *rewrite closure* of I consists of I and the set of all atoms $u\,S\,v$ that have a rewrite proof in I.

A proof $u \Leftarrow_I^R s \Rightarrow_I^S t$ of $u\,T\,t$ is called a *peak*. A proof $u \Leftarrow_I^R t \Leftrightarrow_I^S t$ or $u \Leftrightarrow_I^R u \Rightarrow_I^S t$ of $u\,T\,t$ is called a *plateau*. The peak or plateau is said to be *convergent*, if there exists a rewrite proof of $u\,T\,t$ in I. I is said to be *complete* if all peaks and plateaus in I converge. A straightforward generalization of Proposition 3 is the following completeness criterion:

Proposition 4 *The following three properties are equivalent:*
(i) I is complete.
(ii) Any ground atom which follows from I and Co *has a rewrite proof in I.*
(iii) The rewrite closure of I satisfies Co.

Proof. We prove that (i) implies (ii), the other implications being trivial. Proofs which are not rewrite proofs will be transformed into smaller proofs where the proof ordering is the multiset extension of $\succ$ applied to the terms that occur in the proof. The transformations will be of a local nature which is made possible by the associativity of the composition laws.

Let I commute and let $u = u_0\,S_1\,u_1\,S_2\,u_2 \ldots u_{n-1}\,S_n\,u_n = v$ be a proof of $u\,S\,v$ in I. If $n = 1$ the proof is a rewrite proof. If $n > 1$ and if the proof is not a rewrite proof then it contains a subproof of the form $w\,U\,s\,V\,t$, with $s \succeq w$, $s \succeq t$, and $U \circ V \subseteq T$ is in Co, with either $T = U$ or $T = V$. If the subproof is a peak or plateau, we may replace it by a rewrite proof for $w\,T\,t$ in I which, according to the assumption, exists. Otherwise $s = t = w$. In that case one of the two steps is of the form $s\,T\,s$, and we may simply delete the other. The new proof proves $u\,S\,v$ and is smaller than the given proof. $\Box$

This proof entirely follows standard methods. We have given it in detail to indicate where we make use of the the fact that in composition laws the relation on the right side must be one of the operand relations of the left side. To relax that restriction one will have to treat the reflexive parts of relations more specifically.

From now on we also consider symmetry axioms

$$x\,R\,y \rightarrow y\,R\,x$$

for special relations and in that case call R *symmetric*. Symmetry will be built into the notation in that we assume that an atomic expression $s\,R\,t$ denotes both $s\,R\,t$ and $t\,R\,s$. We require that if U is symmetric and $S \circ T \subseteq U$ then both S and T are also symmetric. We use L to denote the union of the composition and symmetry laws.

Subsequently we restrict ourselves to the case where all predicate symbols are binary. Any other atom $p(t_1, \ldots, t_n)$ can be represented (in a many-sorted framework) by an equivalence $p(t_1, \ldots, t_n) \equiv \top$ where $\equiv$ is an equivalence relation (cf. section 5.1 for more details). Equivalence relations will be investigated more specifically in section 5.

4.2 Redundancy

We sketch the main ideas of an abstract notion of redundancy and refer to Bachmair and Ganzinger (1994b) for further details. Let N and S be sets of clauses and C be a ground clause (not necessarily a ground instance of N). Our intention is to specify when C is not needed for proving the potential inconsistency of $N \cup S$. S will usually refer to clauses, such as the composition laws, which are built into the chaining inference system.

Redundancy is a notion which exploits certain properties of our model construction, hence depends on the given clause ordering. For the refined chaining calculus we need to refine the complexity measure c_L for literals L to become compatible with symmetry: If L is a ground literal A or $\neg A$, with A an atom of the form $u\,S\,v$, then $c_L = (max_L, p_L, s_L)$ where (i) max_L is the maximal term among u and v; (ii) p_L is 1, if L is negative, and 0 otherwise; and (iii) s_L is 1 if S is not symmetric and if $u \succeq v$, and s_L is 0, otherwise.

Again, an admissible ordering on ground clauses is the multiset extension of an admissible (with respect to the refined complexity measure) literal ordering, and we use $\succ$ to denote both the term and literal ordering as well as its multiset extension to an ordering on clauses.

From the proof of the Lemma 3 we observe that inferences were only required when they involve ground clauses C that are false in I_C and in situations where one could assume that all smaller clauses are true in I_C. The following definition of redundancy for clauses approximates these properties without making any explicit reference to the model construction. We call C *S-redundant* with respect to N if there exist ground instances $C_1, \ldots, C_k$ of N such that C is true in every S-model of $C_1, \ldots, C_k$ and $C \succ C_j$, for all j with $1 \leq j \leq k$. It can easily be seen that the clauses $C_1, \ldots, C_k$ can be assumed to be non-redundant. Clearly, if the C_i are true in I_C, C cannot be false in I_C, provided I_C satisfies S. A non-ground clause is called *S-redundant* if all its ground instances are.

Tautologies are S-redundant in this sense, for any S, and most cases of proper subsumption are also covered by this notion of redundancy. The axioms in S are all S-redundant by definition.

Further inspection of the proof of Lemma 3 reveals that, for binary inferences, the first premise is always a productive clause C. Its maximal literal A is strictly maximal, positive, and gets produced into I^C. While the ordering constraints for productivity are captured with the ordering constraints for the inferences, the fact that A is true in I^C will be reflected by the subsequent definition of redundancy for inferences. To that end, we call the maximal atom of the first premise of a binary inference the *main atom* of the inference. For other types of inferences, the main atom is defined to be $\top$. With these preliminaries, a ground inference with conclusion B, maximal premise C, and main atom A is called *S-redundant* with respect to N if either some premise is redundant, or else there exist ground instances $C_1, \ldots, C_k$ of N such that B is true in every S-model of $A, C_1, \ldots, C_k$, and $C \succ C_j$, for all j with $1 \leq j \leq k$. A non-ground inference is called S-redundant if all its ground instances (that satisfy the associated restrictions) are redundant.

We say that a set of clauses N is *saturated up to S-redundancy* with respect to some inference system, if all inferences from N are S-redundant. For the remainder of this paper "redundancy" is meant to refer to L-redundancy whenever the concrete set of built-in axioms L is clear from the context.

For inference systems in which the first premise of a binary ground inference is smaller than the second, the the definition of redundancy given here is equivalent to notions of redundancy that we have proposed in previous work (Bachmair and Ganzinger 1990, Bachmair and Ganzinger 1994b). With the present definition we may admit larger classes of clause orderings.

For a refutationally complete inference system we expect that if N is saturated up to redundancy then N is unsatisfiable if and only if the empty clause is in N. That leaves us with the problem of how to effectively saturate a set of clauses. If the system is *monotone* in the sense that the conclusion of any ground inference is smaller than the maximal premise of the inference, then one way to render an inference redundant is to add its conclusion to N. Given a recursive and monotone inference system, for any recursively enumerable set of clauses an equivalent and saturated set can be recursively enumerated. The inference systems that we present below will usually not be monotone for arbitrarily chosen admissible clause orderings. However, in all cases it will not be difficult, when supplied with any admissible term ordering, to extend it to an admissible clause ordering for which the inferences are monotone.

4.3 Ordered Chaining for General Composition Laws

The inference system below represents a generalization of ordered chaining to general composition laws and to the case of general clauses. We assume an admissible ordering $\succ$ to be given.

Ordered Chaining:

$$\frac{C\,,\,u\,S\,s \quad D\,,\,t\,T\,v}{C\sigma\,,\,D\sigma\,,\,u\sigma\,U\,v\sigma}$$

where (i) $S \circ T \subseteq U$ is in Co; (ii) σ is the most general unifier of s and t; (iii) $u\sigma\,S\,s\sigma$ is strictly maximal with respect to $C\sigma$; (iv) $t\sigma\,T\,v\sigma$ is strictly maximal with respect to $D\sigma$; (v) $u\sigma \not\succeq s\sigma$, and $u\sigma \neq s\sigma$ if $U = T$; and (vi) $v\sigma \not\succeq t\sigma$, and $t\sigma \neq v\sigma$ if $U = S$.

Negative Chaining:

$$\frac{C\,,\,\neg(u\,S\,s) \quad D\,,\,t\,T\,v}{C\sigma\,,\,D\sigma\,,\,\neg(v\sigma\,U\,s\sigma)}$$

where (i) $T \circ U \subseteq S$ is in Co; (ii) σ is the most general unifier of u and t; (iii) $\neg(u\sigma\,S\,s\sigma)$ is maximal with respect to $C\sigma$; (iv) $t\sigma\,T\,v\sigma$ is strictly maximal with respect to $D\sigma$; (v) $v\sigma \not\succeq t\sigma$, and (vi) $s\sigma \not\succeq u\sigma$; and

$$\frac{C\,,\,\neg(u\,S\,s) \quad D\,,\,t\,T\,v}{C\sigma\,,\,D\sigma\,,\,\neg(u\sigma\,U\,t\sigma)}$$

where (i) $U \circ T \subseteq S$ is in Co; (ii) σ is the most general unifier of s and v; (iii) $\neg(u\sigma\,S\,s\sigma)$ is maximal with respect to $C\sigma$; (iv) $t\sigma\,T\,v\sigma$ is strictly maximal with respect to $D\sigma$; (v) $u\sigma \not\succ s\sigma$, and (vi) $t\sigma \not\succeq v\sigma$.

Ordered Resolution:

$$\frac{C\,,\,A \quad D\,,\,\neg B}{C\sigma\,,\,D\sigma}$$

where (i) σ is the most general unifier of A and B; (ii) $A\sigma$ is strictly maximal with respect to $C\sigma$; and (iii) $B\sigma$ is maximal with respect to $D\sigma$.

Ordered Factoring:

$$\frac{C\,,\,A\,,\,B}{C\sigma\,,\,A\sigma}$$

where (i) σ is the most general unifier of A and B; and (ii) $A\sigma$ is maximal with respect to $C\sigma$.

Unfortunately, it also turns out that for disjunctions of binary atoms with multiple occurrences of the maximal term of a clause the inference mechanism of negative and ordered chaining is insufficient. For example, take the set of clauses

$$\begin{array}{lrcl}
(1) & & \rightarrow & c \leq b \\
(2) & & \rightarrow & b \leq c \\
(3) & & \rightarrow & a \leq b\,,\, a \leq c \\
(4) & a \leq b\,,\, a \leq c & \rightarrow &
\end{array}$$

where $\leq$ is transitive and $a \succ b \succ c$. This set of clauses is unsatisfiable. The third clause implies that $a \leq b$ or $a \leq c$ is true. But by the first two clauses, if one of the two inequalities is true, so is the other. This contradicts the last clause. From the first two clauses we obtain $c \leq c$ by chaining. From the last two clauses we obtain $a \not\leq c \vee a \leq c$ by ordered resolution. This inference is redundant, as its conclusion is a tautology. Thus, by adding $c \leq c$ the set is saturated up to redundancy, but does not contain the empty clause.

The following inference rules restore refutational completeness and guarantee compatibility with redundancy.

Composition Resolution:

$$\frac{C\,,\,u\,V\,v\,,\,u'\,U\,v' \quad D\,,\,s\,S\,t}{D\sigma\,,\,\neg(t\sigma\,T\,v'\sigma)\,,\,u'\sigma\,U\,v'\sigma}$$

where (i) $S \circ T \subseteq U$ is in Co; (ii) σ is the most general unifier of u, u' and s; (iii) the literal $u\sigma\,V\,v\sigma$ is strictly maximal with respect to $C\sigma \vee u'\sigma\,U\,v'\sigma$; (iv) the literal $s\sigma\,S\,t\sigma$ is strictly maximal with respect to $D\sigma$; and (v) $v\sigma \not\succ u\sigma$, $v'\sigma \not\succ u\sigma$, $t\sigma \not\succeq s\sigma$, $u'\sigma\,U\,v'\sigma \not\succeq u\sigma\,V\,v\sigma$ and $u\sigma\,V\,v\sigma \not\succ s\sigma\,S\,t\sigma$.

$$\frac{C\,,\,u\,V\,v\,,\,u'\,U\,v' \quad D\,,\,s\,S\,t}{D\sigma\,,\,\neg(u'\sigma\,T\,s\sigma)\,,\,u'\sigma\,U\,v'\sigma}$$

where (i) $T \circ S \subseteq U$ is in Co; (ii) σ is the most general unifier of v, v' and t; (iii) the literal $u\sigma\,V\,v\sigma$ is strictly maximal with respect to $C\sigma \vee u'\sigma\,U\,v'\sigma$; (iv) the literal $s\sigma\,S\,t\sigma$ is strictly maximal with respect to $D\sigma$; and (v) $u\sigma \not\succ v\sigma$, $u'\sigma \not\succ v\sigma$, $s\sigma \not\succeq t\sigma$, $u'\sigma\,U\,v'\sigma \not\succeq u\sigma\,V\,v\sigma$ and $u\sigma\,V\,v\sigma \not\succ s\sigma\,S\,t\sigma$.

In the above example, the third clause resolves with itself by composition resolution yielding

$$(5) \qquad b \leq c \;\rightarrow\; a \leq c$$

Now the refutation proceeds by:

$$\begin{array}{lrcll}
(6) & b \leq c,\, c \leq b,\, a \leq c & \rightarrow & & \text{[neg. chaining of (5) into (3)]} \\
(7) & b \leq c,\, b \leq c,\, c \leq b,\, c \leq c & \rightarrow & & \text{[neg. chaining of (5) into (6)]} \\
(8) & c \leq c & \rightarrow & & \text{[resolution of (1) \& (2) with (7)]} \\
(9) & & \rightarrow & c \leq c & \text{[chaining of (1) and (2)]} \\
(10) & & \rightarrow & & \text{[resolution of (8) with (9)]}
\end{array}$$

Composition resolution represents the controlled application of resolution with the composition laws. For instance, consider the case of a transitive relation S. The clause $D\sigma \vee \neg(t\sigma\, S\, v'\sigma) \vee s\sigma\, S\, v'\sigma$ is an instance of the conclusion of the resolution inference

$$\frac{D\sigma\,,\, s\sigma\, S\, t\sigma \quad \neg(x\, S\, y)\,,\, \neg(y\, S\, z)\,,\, xSz}{D\sigma\,,\, \neg(t\sigma\, S\, z)\,,\, s\sigma\, S\, z}$$

In other words, the conclusion of a composition resolution inference π is an instance of a resolvent between the second premise of π and a composition law in Co. The first premise of π (that is, the existence of a clause $C\,,\, u\, S\, v\,,\, u'\, S\, v'$ with $\sigma = \mathsf{mgu}(s, u, u')$ in the example) regulates which instances and resolvents are needed. We shall later see that in the presence of additional logical properties (eg., symmetry or totality) this inference can be drastically simplified.

The calculus of all these inference rules is denoted by $\mathsf{C}^{\succ}_{\mathsf{L}}$, or simply $\mathsf{C}^{\succ}$ or C. The soundness of the inferences can be checked easily.

It is not difficult to construct admissible clause orderings for which $\mathsf{C}^{\succ}$ is monotone. Let $\succ$ be some total and well-founded ordering on ground terms. Suppose that we extend the complexity triples c_L for ground literals L to quadrupels $c_L = (max_L, p_L, s_L, d_L)$ where the additional component d_L is 1 if the maximal term of L occurs twice in L, and 0, otherwise. Then any total, well-founded ordering $\succ$ on literals compatible with c_L and its multiset extension to clauses will do.

Lemma 4 *Let $\succ$ be of the above form. Then C is monotone.*

Proof. The lemma follows by straightforward inspection of the various cases. Consider, for instance, a ground instance of (the first variant of) composition resolution. It has the form

$$\frac{C\,,\, s\, V\, v\,,\, s\, U\, v' \quad D\,,\, s\, S\, t}{D\,,\, \neg(t\, T\, v')\,,\, s\, U\, v'}$$

where $s\, V\, v$ is strictly maximal in $C \vee s\, S\, v'$; the literal $s\, S\, t$ is strictly maximal in D; and $s \succeq v$, $s \succeq v'$, $s \succ t$, and $(s\, S\, t) \succeq (s\, V\, v) \succ (s\, U\, v')$. From the new component d_L in the literal ordering and as U is not symmetric we may infer that $(s\, S\, t) \succ (s\, U\, v')$ is impossible if $s = v'$. Hence $s \succ v'$ and therefore $(s\, V\, v) \succ \neg(t\, T\, v')$. The conclusion is therefore smaller than the second premise. □

4.4 Refutational Completeness

Referring to the model construction of Section 3.3., we first state the main lemma.

Lemma 5 *Let N be a set of clauses that is saturated up to redundancy with respect to $\mathsf{C}^{\succ}$ and does not contain the empty clause. Furthermore, let I be the interpretation constructed from the ground instances of N according to $\succ$. Then for every ground instance C of a clause in N we have:*

1. *If C is productive then it is non-redundant.*
2. *If D is a clause $D' \vee A$ which produces A and if $D \preceq C$ then D' is false in I^C.*
3. *I_C as well as I^C satisfy L.*
4. *The clause C is true in I^C.*

Proof. We use induction on $\succ$. Let C be a ground instance of N such that (1)–(4) are satisfied for all smaller ground instances of N.

(1) Let C be L-redundant in N. Then there exist ground instances $C_1, \ldots, C_n$ of N such that C logically follows from the C_i, that is, is true in every L-model of the C_i. Applying the induction hypothesis of (3) and (4) we infer that I_C is an L-model of the C_i. Therefore C is true in I_C and, hence, not productive.

(2) Suppose D is a productive clause of the form $D' \vee A$ with $A \succ D'$ and $C \succeq D$. We prove, by means of induction, that D' is false in $I^{C'}$, for any clause $C \succeq C' \succeq D$. As induction hypothesis we may assume that D' is false in $I_{C'}$. (This is easily confirmed for the base case $C' = D$.) Suppose D' is true in the rewrite closure of $R^{C'}$. Let s be the maximal term in D. If $\neg B$ is a negative literal in D' then B is true in $I_{C'}$ and hence true in $I^{C'}$. If B is a positive literal in D' then B is false in $I_{C'}$, and hence does not have a rewrite proof in $R_{C'}$. If s does not occur in B then C' cannot produce an atom that would be applicable in a rewite proof of B, hence B is false in $I^{C'}$. There are two remaining cases to be considered. One is that C' is of the form $C'' \vee s\,S\,t$ and produces $s\,S\,t$ with $s \succeq t$; and moreover A is an atom $s\,V\,v$ and D' is of the form $D'' \vee s\,U\,v'$, such that $s\,U\,v'$ can be proved by a rewrite proof consisting of $s\,S\,t$, followed by a rewrite proof for some $t\,T\,v'$ in $R_{C'}$, with $S \circ T \subseteq U$. Hence $s \succ t$. As both C' and D are productive, by (1) they are not redundant. Since N is saturated up to redundancy, the clause $\hat{C} = C'' \vee \neg(t\,T\,v') \vee s\,U\,v'$ that is obtained by composition resolution from D and C' follows from A and from instances of N which are smaller than C'. A, as well as, by induction hypothesis for (4), the C_i, are true in $I_{C'}$, and we may infer that $\hat{C}$ is true in $I_{C'}$. Also, since C' is productive, C'' is false in $I_{C'}$. Therefore, $s\,U\,v'$ must be true in $I_{C'}$. This contradicts our assumption that D' is false in $I_{C'}$. The other, similar case that C' is of the form $C'' \vee s\,S\,t$ and produces $s\,S\,t$ with $t \succ s$, corresponds to the other version of composition resolution. Details are left to the reader.

(3) By Proposition 4 we have to show that both R_C and R^C are complete. By the induction hypothesis, peaks and plateaus that involve only R_C converge. The assertion for I^C is thus trivially true if E_C is empty, or if $E_C = \{t\,T\,v\}$ where $v \succ t$ and T is not symmetric. Suppose $C = C' \vee v\,T\,t$ (modulo the potential symmetry of T) produces $v\,T\,t$ with $v \succ t$. By the definition of E_C, the clause C' is false in I_C. If there is another clause $D = D' \vee u\,S\,v$ with $C \succ D$ that produces $u\,S\,v$ with $v \succ u$, and $S \circ T \subseteq U$, then $u \Leftarrow^S v \Rightarrow^T t$ is a peak in I^C. (If $v = u$ and $S = U$ then $u \Leftrightarrow^S v \Rightarrow^T t$ is a plateau in I^C.) Since $C \succ D$ we may apply (2) to conclude that D' is also false in I_C. Furthermore, C and D are by (1) non-redundant and produce $C'' = C' \vee D' \vee u\,U\,t$ by ordered chaining. Since N is saturated up to redundancy, the clause C'' must logically follow from $u\,S\,v$ and from clauses strictly smaller than C. That is, C'' must be true in I_C, but $C' \vee D'$ is false in I_C. Therefore $u\,U\,t$ is true in I_C, hence $u \Downarrow^U_{I_C} t$, which establishes the required commutation property. If $C = C' \vee v\,T\,v$ produces $v\,T\,v$ then there might be a plateau in I^C due to the existence of a clause $D \prec C$ producing an atom $u\,S\,v$ [or $v\,S\,u$], with $v \succ u$. That $u\,U\,v$ [$v\,U\,u$] is true in I_C is proved in a similar way.

(4) We already know that all ground instances of N that are smaller than C are true in I_C and that I_C is a model L. If C is redundant, then it has to be true in I_C and in hence in I^C. If C is productive, it is by construction true in the rewrite closure of R^C. Let C be a non-redundant, non-productive clause.

(4.1) First suppose the maximal literal in C is negative and C can be written as $C' \vee \neg A$, with a maximal atom A that is true in I_C.

If A is produced by some (smaller) clause $D = D' \vee A$, then the resolvent $C'' = D' \vee C'$ of C and D is smaller than C (since $\neg A \succ D'$). Since N is saturated up to redundancy, C'' must follow from clauses smaller than C, all of which are true in I_C. Therefore C'' must be true in I_C. By (2), D' is false in I_C. Therefore C', and also C, is true in I_C.

If A is a binary atom $u\,U\,v$ with a rewrite proof $u \Downarrow^U_{R_C} v$ of length ≥ 2, then there exists a suitable productive clause $D' \vee u\,S\,v'$ (where $v' \Downarrow^T_{R_C} v$) or $D' \vee u'\,T\,v$ (where $u \Downarrow^S_{R_C} u'$) and $S \circ T \subseteq U$. By negative chaining we get either $C' \vee D' \vee \neg(v'\,T\,v)$ or $C' \vee D' \vee \neg(u\,S\,u')$. In either case we may use saturation up to redundancy to infer that C', and hence C, is true in I_C.

(4.2) If C is of the form $C' \vee A \vee A$, with multiple occurrences of the maximal literal A, then the smaller clause $C' \vee A$ is obtained from C by ordered factoring. Using the induction hypothesis and saturation up to redundancy, we may infer that $C' \vee A$ and C must be true in I_C.

Cases (4.1) and (4.2) cover all the possibilities for a non-redundant clause C to be non-productive. In each case we have shown C to be true in I_C, which completes this case. $\Box$

As an immediate corollary of the above lemma we obtain the following completeness theorem.

Theorem 2 *If a set of clauses N is saturated up to redundancy with respect to C_L, then the set $N \cup \mathsf{L}$ is unsatisfiable if and only if it contains the empty clause.*

5 Partial Congruences

5.1 Replacement Axioms

In this section we look at replacement axioms of the form

$$x\,S\,y,\ P[x] \to P[y]$$

which express that whenever $x\,S\,y$ then P remains true if we replace the indicated occurrence of x in P by y. For example, if S is equality then this law is valid for arbitrary formulas P and occurrences x. Our intention is to represent this operational meaning in a chaining calculus in which chaining through subterms such as x in $P[x]$ is performed in the presence of binary atoms such as $x\,S\,y$. The main problem with subterm chaining is that in general we cannot avoid chaining below variables. For example, if we have the axioms

$$\begin{aligned} x < y,\ y + z < u &\ \to\ x + z < u \\ x < y,\ z + y < u &\ \to\ z + x < u \end{aligned}$$

expressing the monotonicity of an operation $+$ in both arguments with respect to $<$, then, given inequations $3 < 5$ and $x+y < y+17$, we would have to infer that $x+(u+3) < (u+5)+17$ where this inequation is obtained by chaining with $3 < 5$ below the variable y in the second inequation, instantiating it by $u+5$. This problem is discussed in more detail in Section 5.7. Chaining below variables is unfeasable in practice, and nothing is gained compared to a strategy by which monotonicity axioms are treated unspecifically just like any other clause. We will develop subterm chaining only for a class of binary relations for which variable overlaps are generally redundant. These relations will have to be symmetric, in particular. Orderings are not symmetric and hence the results of this paper will not allow to treat monotonicity laws for orderings by subterm chaining.

The relations that we will be able to handle are partial congruences. These are partial equivalences which in addition allow the replacement of equals by equals in arbitrary contexts. A *partial equivalence* $\equiv$ is a binary relation which is symmetric, transitive and satisfies $\equiv \circ S \subseteq S$ and $S \circ \equiv \subseteq S$, for any other binary relation S. A partial equivalence need not be reflexive. Reflexivity properties may, for instance, be inferred through the implication

$$x \sim y \rightarrow x \sim x$$

which is a consequence of transitivity and symmetry. We call a term *defined* with respect to $\sim$ if $t \sim t$ can be derived, and the set of all defined ground terms is called the *domain* of the partial equivalence.

A *partial congruence* $\sim$ is a symmetric relation satisfying the replacement axiom

$$x \sim y,\ A[x] \quad \rightarrow \quad A[y]$$

for any atom A and any occurrence of x in A. Partial congruences are special cases of partial equivalences. A partial congruence need not satisfy any of the *monotonicity laws*

$$x \sim y \rightarrow\ f(\ldots x \ldots) \sim f(\ldots y \ldots).$$

Monotonicity (together with symmetry and transitivity) implies replacement, but not vice versa.

Partial congruences are useful for axiomatizing partial functions. They occur in Scott's logic of partial equality (Scott 1979) and also in type theory where $t \approx_T t$ in a type T can be proven only if t is provably a member of T.

Example. A specification of natural numbers with a partial predecessor function might for instance be given as follows:

$$\begin{array}{rcl} nat(x) & \leftrightarrow & x \sim x \\ & \rightarrow & nat(0) \\ nat(x) & \leftrightarrow & nat(s(x)) \\ 0 \sim s(x) & \rightarrow & \\ nat(x) & \rightarrow & p(s(x)) \sim x \\ nat(p(x)) & \rightarrow & s(p(x)) \sim x \end{array}$$

The undefinedness of $p(0)$ is represented by the fact that $\neg nat(p(0))$ can be derived as a theorem. $\Box$

An example of a partial congruence for which the replacement axioms can be represented finitely (for finite signatures) is equality for strict partial functions:

Example.

$$\begin{array}{rcl} x \sim y & \rightarrow & y \sim x \\ x \sim y,\, y \sim z & \rightarrow & x \sim z \\ f(x_1, \ldots, x_n) \sim f(x_1, \ldots, x_n) & \rightarrow & x_i \sim x_i, \text{ for any } f \text{ and position } i \\ x_i \sim y_i\,,\, f(\ldots x_i \ldots) \sim z & \rightarrow & f(\ldots y_i \ldots) \sim z, \text{ for any } f \text{ and } i \end{array}$$

It is easy to see that these axioms in fact imply the replacement axioms. The third axiom (scheme) specifies strictness of all functions. □

Congruences $\approx$ are partial congruences which are reflexive. The replacement laws are then equivalent to the monotonicity laws

$$x \approx y \rightarrow \ f(\ldots x \ldots) \approx f(\ldots y \ldots), \text{ for any function } f.$$

together with reflexivity, symmetry and transitivity. Hence, if the signature is finite we again have a finite representation of the replacement laws. Note, however, that for the subsequent exposition of chaining finiteness of the representation of the replacement axioms is inessential.

Example. The clauses

$$\begin{array}{rcl}
x < y\,,\, y < z & \rightarrow & x < z \\
 & & x \approx x \\
x \approx y & \rightarrow & y \approx x \\
x \approx y\,,\, y \approx z & \rightarrow & x \approx z \\
x \approx y & \rightarrow & f(\ldots x \ldots) \approx f(\ldots y \ldots), \text{ for any function } f. \\
x \leq y\,,\, y \leq z & \rightarrow & x \leq z \\
x \approx y\,,\, y < z & \rightarrow & x < z \\
x < y\,,\, y \approx z & \rightarrow & x < z \\
x \approx y\,,\, y \leq z & \rightarrow & x \leq z \\
x \leq y\,,\, y \approx z & \rightarrow & x \leq z \\
x < y\,,\, y \leq z & \rightarrow & x < z \\
x \leq y\,,\, y < z & \rightarrow & x < z \\
x < x & \rightarrow &
\end{array}$$

express the basic properties of equality and of and a strict and a non-strict inequality. The subterm chaining calculus to be presented below will be able to handle them implicitly. No explicit inferences from these axioms will be needed for refutational completeness. □

In a single-sorted framework any two partial congruences must be identical on the intersection of their domains. To simplify the technical presentation we will therefore assume that we have just one partial congruence $\sim$ in addition to equality $\approx$. $\approx$ is in particular the reflexive closure of $\sim$. We write $\equiv$ to denote $\sim$ or $\approx$ and call atoms of the form $s \equiv t$ *equalites*. We also write $s \not\equiv t$ for *negative equalities* $\neg\, s \equiv t$. We assume that the composition laws $\equiv \circ\, S \subseteq S$ and $S \circ \equiv\, \subseteq S$, for any S, are contained in Co, and that Co does not contain any other composition laws involving $\sim$ and $\approx$. We also assume that the symmetry laws for $\sim$ and $\approx$ are in Sy. By L we again denote the set of all laws that we intend to built into the inference system, that is, the union of Co and Sy, together with the replacement axioms for $\sim$ and $\approx$ and the reflexivity axiom for $\approx$.

We extend our previous notion of a proof from Section 4.1. Let I be a set of ground atoms. Remember that symmetry is built into the notation. A *proof* in I is now a sequence of the form

$$u = u_0\, S^0|_{p_0}\, u_1\, S^1|_{p_1}\, u_2 \ldots u_{n-1}\, S^{n-1}|_{p_{n-1}}\, u_n = v$$

where $n \geq 0$, and for each $0 \leq i < n$ there exists $v_i S_i v_{i+1}$ in I such that $u_i|_{p_i} = v_i$, $u_{i+1} = u_i[v_{i+1}]_{p_i}$ and p_i is the root in u_i if S_i is different from $\sim$ and $\approx$. For each i, p_i denotes the term position at which the atom from I that is used in the i-th proof step is applied. Subsequently we will omit positions from the notation for proofs whenever they are irrelevant. The proof is said to *prove* $u \approx v$ if $n = 0$ (and $u = v$) or else if $n > 0$ and each S_i is either $\sim$ or $\approx$. The proof additionally proves $u\,S\,v$ if $S^1 \circ S^2 \circ \ldots \circ S^n \subseteq S$ follows from Co and if there exists at least one $0 \leq i < n$ such that p_i is the root position in u_i and S_i is different from $\approx$. For example if $a \sim b$ and $f(a) \approx f(c)$ are in I then $f(c) \approx f(a) \sim f(b)$ is a proof of $f(c) \approx f(b)$ but it is not a proof of $f(c) \sim f(b)$. On the other hand, if $a \approx b$ and $f(a) \sim f(c)$ are in I then $f(c) \sim f(a) \approx f(b)$ is a proof of both $f(c) \sim f(b)$ and $f(c) \approx f(b)$.

Proposition 5 *Let I be a set of ground atoms and let u and v be ground terms. Then $u\,S\,v$ follows from I and L if and only if there exists a proof of $u\,S\,v$ in I.*

The characteristic properties of a rewrite proof (relative to a given well-founded ordering $\succ$ which is total on ground terms) are as before. More precisely, a proof of the above form is a *rewrite proof* if $n = 0$, or else if $n = 1$, $u = v$ and $S^1 = \sim$, or else if $n \geq 1$ and

$$u = u_0 \succ \ldots \succ u_k \preceq u_{k+1} \prec \ldots \prec u_n = v.$$

Additionally we require for a rewrite proof that S^k be different from $\equiv$ if $n > 0$ and $u_k = u_{k+1}$. then S^k is different from $\equiv$. The *rewrite closure of* I consists of I and the set of all atoms $u\,S\,v$ that have a rewrite proof in I. For an S-step in a rewrite proof over I we shall again use the notations $s \Rightarrow_I^S t$, $s \Leftarrow_I^S t$, or $s \Leftrightarrow_I^S t$, depending on whether $s \succ t$, $s \prec t$ or $s = t$, respectively. More refined notations such as $u[s]_p \Rightarrow_I^{\equiv} u[t]_p$ are used for denoting equality steps whenever the rewrite position needs to be explicitly indicated.

As before, a proof $u \Leftarrow_I^R s \Rightarrow_I^S t$ of $u\,T\,t$ is called a *peak*, and a proof $u \Leftarrow_I^R t \Leftrightarrow_I^S t$ or $u \Leftrightarrow_I^R u \Rightarrow_I^S t$ of $u\,T\,t$ is called a *plateau* in I. The peak or plateau is said to be *convergent*, if there exists a rewrite proof of $u\,T\,t$ in I. I is said to be *complete* if all peaks and plateaus in I converge.

Proposition 6 *Then following three properties are equivalent:*
(i) I is complete.
(ii) Any ground atom which follows from I and L has a rewrite proof in I.
(iii) The rewrite closure of I satisfies L.

For obtaining a workable inference system of ordered subterm chaining, we have to place additional restrictions on the term ordering. The ordering has to be stable under contexts, i.e., should be a rewrite relation. A *rewrite relation* is a binary relation $\succ$ on terms such that $s \succ t$ implies $u[s\sigma] \succ u[t\sigma]$, for all terms u and substitutions σ. A *reduction ordering* is a well-founded rewrite relation. From now on $\succ$ will always denote a *complete* reduction ordering, i.e., a reduction ordering that is total on ground terms.

A peak $u[s]_p \Leftarrow^R u[t]_p = u[s']_q \Rightarrow^S u[t']_q$ is called *critical* if p or q is the root position. A plateau $u[s]_p \Leftrightarrow^R u[s]_p = u[s']_q \Rightarrow^S u[t']_q$ or $u[s]_q \Leftarrow^R u[t]_q = u[s']_p \Leftrightarrow^S u[s']_p$ is called a *critial* if p is the root position and if R is different from $\approx$.

Proposition 7 *I is complete if and only if all critical peaks and plateaus commute.*

5.2 The Many-Sorted Case

Throughout this paper we assume a single-sorted framework with at most one partial congruence (in addition to equality). The extension the case of many-sorted signatures and arbitrary families of partial congruences is a simple exercise that we leave to the reader. In practice a many-sorted framework is desirable. In that context one may want to have a (partial) congruence on each sort. The following example in particular justifies a posteriori why we can safely restrict ourselves to consider binary relations only. Non-binary atoms can be easily modelled in a two-sorted signature with sorts *prop* for (atomic) propositions for predicate symbols that are no special binary relations, and a sort o for objects. Function symbols are of type $o^n \to o$ and non-special predicate symbols are of type $o^n \to prop$. $\approx$ denotes equality on o and $\leftrightarrow$ denotes equality on *prop*. In the following clauses, x, y and z range over o, whereas p, q, r range over *prop*.

$$
\begin{array}{rcl}
 & & x \approx x \\
x \approx y & \to & y \approx x \\
x \approx y\,,\, y \approx z & \to & x \approx z \\
x \approx y & \to & F(\ldots x \ldots) \approx F(\ldots y \ldots), \text{ for any function } F \\
x \approx y & \to & P(\ldots x \ldots) \leftrightarrow P(\ldots y \ldots), \text{ for any predicate } P \\
 & & p \leftrightarrow p \\
p \leftrightarrow q & \to & q \leftrightarrow p \\
p \leftrightarrow q\,,\, q \leftrightarrow r & \to & p \leftrightarrow r
\end{array}
$$

With these axioms in L a user may actually write clauses in which equivalences between propositions occur on literal position in clauses. An example of such an "extended" clause would be

$$P(x) \leftrightarrow Q(s(x)) \to P(p(x)) \leftrightarrow Q(x).$$

In general, ordered chaining for $\leftrightarrow$-atoms is ordered rewriting with equivalences. Orientable equivalences can be used for simplification of atoms by reduction in the same way as one uses orientable equations for the simplification of terms. A (minimal) constant $\top : prop$ can be introduced to express truth of an atom $P(t_1, \ldots, t_n)$ by writing an equivalence $P(t_1, \ldots, t_n) \leftrightarrow \top$. Negative ordered chaining between to equivalences $P \leftrightarrow \top$ corresponds to ordered resolution while positive chaining yields tautologies of the form $C \vee (\top \leftrightarrow \top)$ which are redundant.

5.3 The Subterm Chaining Calculus

Let $\succ$ be a complete reduction ordering on terms. Once more we refine the complexity measure c_L for literals L, so as to accomodate for partial equivalences and congruences. If L is a ground literal A or $\neg A$, with A an atom of the form $u\,S\,v$, then $c_L = (max_L, p_L, s_L)$, where (i) p_L is 1, if L is negative, and 0 otherwise; (ii) max_L is the maximal term of u and v; s_L is 0, if S is $\sim$ or $\approx$; otherwise, if S is different from $\sim$ and $\approx$, then s_L is 2, if $u \succeq v$ and S is not symmetric, and s_L is 1, otherwise. We observe that $\equiv$-literals are made less complex than other literals with the same maximal term. That will have the effect that equalities are defined at an earlier stage in the model construction. An admissible ordering on ground clauses is the multiset extension of an admissible ordering on literals (based on the refined complexity measure), and the extensions to non-ground expressions is as above.

Ordered Subterm Chaining:

$$\frac{C\,,\, u\, S\, s \quad D\,,\, t\,[\,s']_p T\, v}{C\sigma\,,\, D\sigma\,,\, t[u]_p\sigma\, U\, v\sigma}$$

where (i) σ is the most general unifier of s and s'; (ii) if p is the root in t then $S \circ T \subseteq U$ is in Co, and if p is not the root in t then S is $\equiv$ and $U = T$; (iii) $u\sigma\, S\, s\sigma$ is strictly maximal with respect to $C\sigma$; (iv) $t\sigma\, T\, v\sigma$ is strictly maximal with respect to $D\sigma$; (v) $u\sigma \not\succ s\sigma$, and $u\sigma \neq s\sigma$ if $U = T$ or $S = \equiv$; (vi) $v\sigma \not\succ t\sigma$, and $t\sigma \neq v\sigma$ if $U = S$ and p is the root position; and (vii) if S is $\equiv$, then s' is not a variable.

Negative Subterm Chaining:

$$\frac{C\,,\, t\, T\, v \quad D,\, \neg(u[t']_p\, S\, s)}{C\sigma\,,\, D\sigma\,,\, \neg(u[v]_p\sigma\, U\, s\sigma)}$$

where (i) σ is the most general unifier of t and t'; (ii) if p is the root in u, then $T \circ U \subseteq S$ is in L, and if p is not the root in u, then $U = S$ and T is $\equiv$; (iii) $\neg(u\sigma\, S\, s\sigma)$ is maximal with respect to $D\sigma$; (iv) $t\sigma\, T\, v\sigma$ is strictly maximal with respect to $C\sigma$; (v) $v\sigma \not\succeq t\sigma$ and $s\sigma \not\succeq u\sigma$; and (vi) if T is $\equiv$ then t' is not a variable;

and

$$\frac{C\,,\, t\, T\, v \quad D,\, \neg(u\, S\, s[v']_p)}{C\sigma\,,\, D\sigma\,,\, \neg(u\sigma\, U\, s[t]_p\sigma)}$$

where (i) σ is the most general unifier of v and v'; (ii) if p is the root in s, then $U \circ T \subseteq S$ is in L, and if p is not the root in s, then $U = S$ and T is $\equiv$; (iii) $\neg(u\sigma\, S\, s\sigma)$ is maximal with respect to $D\sigma$; (iv) $t\sigma\, T\, v\sigma$ is strictly maximal with respect to $C\sigma$; (v) $u\sigma \not\succ s\sigma$ and $t\sigma \not\succeq v\sigma$; and (vi) if T is $\equiv$ then v' is not a variable.

Ordered Resolution:

$$\frac{C\,,\, u'\, S\, v \quad D,\, \neg\neg(u\, S\, v)}{C\sigma\,,\, D\sigma}$$

where (i) σ is the most general unifier of u, u', v and v'; (ii) $u\sigma\, S\, v\sigma$ is strictly maximal with respect to $C\sigma$; and is maximal with respect to $D\sigma$; and (iii) S is different from $\equiv$.

Reflexivity Resolution:

$$\frac{C\,,\, u \not\approx v}{C\sigma}$$

where (i) σ is the most general unifier of u and v, and (ii) $u\sigma \not\approx v\sigma$ is maximal with respect to $C\sigma$.

Ordered Factoring:

$$\frac{C\,,\, A\,,\, B}{C\sigma\,,\, A\sigma}$$

where (i) σ is the most general unifier of A and B; and (ii) $A\sigma$ is maximal with respect to $C\sigma$.

Composition Resolution: This inference is the same as above, except that it is not required when U is $\equiv$:

$$\frac{C\,,\, u\,V\,v\,,\, u'\,U\,v' \quad D\,,\, s\,S\,t}{D\sigma\,,\, \neg(t\sigma\,T\,v'\sigma)\,,\, u'\sigma\,U\,v'\sigma}$$

where (i) $S \circ T \subseteq U$ is in Co and U is not $\equiv$; (ii) σ is the most general unifier of u, u' and s; (iii) the literal $u\sigma\,V\,v\sigma$ is strictly maximal with respect to $C\sigma \vee u'\sigma\,U\,v'\sigma$; (iv) the literal $s\sigma\,S\,t\sigma$ is strictly maximal with respect to $D\sigma$; and (v) $v\sigma \not\succ u\sigma$, $v'\sigma \not\succ u\sigma$, $t\sigma \not\succeq s\sigma$, $u'\sigma\,U\,v'\sigma \not\succeq u\sigma\,V\,v\sigma$ and $u\sigma\,V\,v\sigma \not\succ s\sigma\,S\,t\sigma$.

$$\frac{C\,,\, u\,V\,v\,,\, u'\,U\,v' \quad D\,,\, s\,S\,t}{D\sigma\,,\, \neg(u'\sigma\,T\,s\sigma)\,,\, u'\sigma\,U\,v'\sigma}$$

where (i) $T \circ S \subseteq U$ is in Co and U is not $\equiv$; (ii) σ is the most general unifier of v, v' and t; (iii) the literal $u\sigma\,V\,v\sigma$ is strictly maximal with respect to $C\sigma \vee u'\sigma\,U\,v'\sigma$; (iv) the literal $s\sigma\,S\,t\sigma$ is strictly maximal with respect to $D\sigma$; and (v) $u\sigma \not\succ v\sigma$, $u'\sigma \not\succ v\sigma$, $s\sigma \not\succeq t\sigma$, $u'\sigma\,U\,v'\sigma \not\succeq u\sigma\,V\,v\sigma$ and $u\sigma\,V\,v\sigma \not\succ s\sigma\,S\,t\sigma$.

Note that if U is different from $\equiv$ neither V nor S can be $\equiv$ if the ordering restrictions are satisfied. That is, we do not require composition resolution for partial congruences. The latter are dealt with by a specialized form of composition resolution, called equality factoring.

Equality Factoring:

$$\frac{C\,,\, u \equiv_1 v\,,\, u' \equiv_2 v'}{C\sigma\,,\, v\sigma \not\equiv_3 v'\sigma\,,\, u\sigma \equiv_2 v'\sigma}$$

where (i) σ is the most general unifier of u and u'; (ii) either $\equiv_2 \,=\, \equiv_3 \,=\, \approx$ or else $\equiv_2$ is $\sim$ and exactly one of the $\equiv_1$ and $\equiv_3$ is $\sim$; (iii) $v\sigma \not\succeq u\sigma$, $v'\sigma \not\succeq u\sigma$, and $v'\sigma \not\succeq v\sigma$; and (iv) $u\sigma \equiv v\sigma$ is strictly maximal with respect to $C\sigma \vee u'\sigma \equiv v'\sigma$.

Equality factoring corresponds to composition resolution of a (suitable) clause with itself.

The subterm chaining calculus will be denoted by $\mathsf{S}^{\succ}$, or simply S.

Remarks. Let us point out the major differences between C and S which are mainly in the treatment of negative literals. A common case of negative chaining is

$$\frac{C\,,\, t\,S\,v \quad D\,,\, \neg(u\,S\,s)}{C\sigma\,,\, D\sigma\,,\, v\sigma \not\approx s\sigma}$$

with $\sigma = \mathsf{mgu}(t,u)$, as $\approx \circ S \subseteq S$ for any relation S. If $v\sigma$ and $s\sigma$ are unifiable (by τ), then from the latter we may infer $C\sigma\tau \vee D\sigma\tau$ by reflexivity resolution. While in the system C the last step of a rewrite proof for a negative literal is always built by a step of ordered resolution, in S it is usually a trivial proof of equality. Ordered resolution now only needs to take care of inequations of the form $t\,S\,t$ with S different from $\equiv$. Non-trivial proofs of equality are obtained by applying facts about congruences in negative subterm chaining inferences of the form

$$\frac{C\,,\, a \equiv b \quad D\,,\, w[b'] \not\approx c}{C\rho\,,\, D\rho\,,\, w[a]\rho \not\approx c\rho}.$$

Consider, for instance, the two clauses

$$\begin{array}{rl} & \rightarrow \quad a \sim b \\ f(a) \approx f(b) & \rightarrow \end{array}$$

with $a \succ b$. The refutation of this example proceeds as follows:

$$\begin{array}{rl} f(b) \approx f(b) & \rightarrow \\ & \rightarrow \end{array}$$

with a step of negative chaining of the first clause into the second clause, followed by an application of equality resolution. Note that $f(b) \not\sim f(b)$ cannot be inferred by negative chaining into the second clause.

More Restrictions. The inference system can be further optimized by adding more restrictions to the inference rules. Certain ground inferences appear as instances of more than one type of non-ground inference, and one may want to avoid that they are computed more than once. Moreover, chaining inferences of the form

$$\frac{C, s \sim u \quad D, s \approx v}{C, D, u \approx v}$$

which are a consequence of $\sim \circ \approx \subseteq \approx$ are redundant given the fact that in this case

$$\frac{C, s \sim u \quad D, s \approx v}{C, D, u \sim v}$$

is another instance of ordered chaining from the same premises, and $\sim \subseteq \approx$ holds for any partial equivalence $\sim$. A dual situation arises in negative chaining.

Monotonicity of Inferences. A clause ordering for which $\mathsf{S}^{\succ}$ is monotone can be constructed as for C, cf. Section 4.3, by adding a component d_L to the complexity measure for literals L that is 1 if the maximal term of L occurs twice in L, and 0, otherwise.

5.4 Lifting

Chaining into or below variables is not needed for equality:

Lemma 6 *Let*

$$\frac{C, s \equiv t \quad D[s]_p}{C, D[t]_p}$$

be a ground inference by ordered or negative subterm chaining. Moreover let D be the ground instance $D'\sigma$ of a clause D' in N such that p is a position at or below a variable in D'. Then the inference is L*-redundant in N.*

Proof. For the inference to be ordered we have $s \succ t$. Suppose $D' = D'[x]_q$ with x a variable and q a position in D' such that $p = q.q'$. Suppose that $x\sigma = s'[s]_{q'}$, and let $t' = s'[t]_{q'}$. Define τ to be a substitution identical to σ except that x is mapped to t'. Then $D'' = D'\tau$ is a ground instance of N for which $D \succ D''$. D'' and $s \equiv t$, together with the laws for (partial) congruences, imply $D[t]$ and, hence, $C \vee D[t]$. (In fact this is the case as partial congruences are symmetric and allow replacement of equals by equals in any context in which x may occur in D'.) Therefore the inference is L-redundant in N. $\square$

For sake of simplicity the lemma does not explicitly refer to those cases of variable chaining with an equality $s \sim t$ in which, in addition to the term replacement, a negative "$\sim$" in the second premise becomes a "$\approx$" or a positive "$\approx$", respectively, becomes a "$\sim$" in the conclusion. Theses inferences are also redundant. The proof is similar and left to the reader.

A consequence of this lemma is that any non-redundant inference from ground instances of N can be obtained as ground instance of an inference from premises in N. In other words, any non-redundant ground inference can be lifted:

Theorem 3 *Let N be a set of clauses that is saturated up to redundancy with respect to S. Then the set of ground instances of N is also saturated up to redundancy with respect to S.*

5.5 Refutational Completeness

With these preliminaries we apply our model construction once more to obtain the following main lemma:

Lemma 7 *Let N be a set of clauses that is saturated up to L-redundancy with respect to S and does not contain the empty clause, and let I be the interpretation constructed from all ground instances of N. Then for every ground instance C of a clause in N we have:*

1. *If C is productive then it is not L-redundant in N.*
2. *If C is a clause $C' \vee u\,S\,v$ (or $C' \vee v\,S\,u$) that produces $u\,S\,v$ (or $v\,S\,u$) with $u \succeq v$, and $D \prec C$ is a clause $D' \vee s \equiv t$ producing $s \equiv t$ with $s \succ t$ then s is different from any subterm of u.*
3. *If D is a clause $D' \vee A$ which produces A and if $D \preceq C$ then D' is false in I^C.*
4. *I^C satisfies L.*
5. *C is true in I^C.*

Proof. We use induction on $\succ$. By Theorem 3 we may assume that the set of ground instances of N is saturated up to redundancy. Let C be a ground clause in N, such that (1)–(5) are satisfied for all smaller clauses in N.

(1) Let C be L-redundant in N. Then there exist ground instances $C_1, \ldots, C_n$ of N such that C logically follows from the C_i, that is, is true in every L-model of the C_i. Applying the induction hypothesis of (4) and (5) we infer that I_C is an L-model of the C_i. Therefore C is true in I_C and, hence, not productive.

(2) Suppose that $C = C' \vee u\,S\,v$ produces $u\,S\,v$ with $u \succeq v$, and $D \prec C$ is a clause $D' \vee s \equiv t$ producing the equation $s \equiv t$ with $s \succ t$. Suppose that $u = u[s]$. The clause $C'' = D' \vee C' \vee u[t]\,S'\,v$ (with either $S' = S$ or else $S' = \sim$, in case $\equiv$ is $\sim$ and S is $\approx$) can be obtained by ordered chaining from C and D. By (1) neither C nor D are redundant. The set of ground instances of N is saturated, hence $D' \vee C' \vee u[t]\,S'\,v$ logically follows from $s \equiv t$ and from instances of N strictly smaller than C. By induction hypothesis (5) the latter are true in I_C, hence C'' is true in I_C. By induction hypothesis for (3), D' is false in I_C. C' is also false in I_C, therefore, $u[t]\,S'\,v$ must be true in I_C. This implies that $u\,S\,v$ is true in I_C, hence C cannot be productive.

(3) Suppose D is a productive clause of the form $D' \vee A$ with $A \succ D'$ and $C \succeq D$. We prove, by means of induction, that D' is false in $I^{C'}$, for any clause $C \succeq C' \succeq D$. As induction hypothesis we may assume that D' is false in $I_{C'}$. (This is easily confirmed

for the base case $C' = D$.) Suppose D' is true in the rewrite closure of $R^{C'}$. There are essentially two cases. One is that C' is of the form $C'' \vee s \equiv t$ and produces an equation $s \equiv t$ with $s \succeq t$; and moreover A is of the form $s\,V\,v$ and D' is of the form $D'' \vee s\,U\,v'$ [or $v'\,U\,s$], such that $s\,U\,v'$ [$v'\,U\,s$] can be proved by a rewrite proof composed by a step of $s \equiv t$, and a rewrite proof for some $t\,T\,v'$ [or $v'\,T\,t$] in $R_{C'}$. Hence $s \succ t$, which implies that $s \succ v'$ and that T, U, and V are $\sim$ or $\approx$. (Otherwise $C' \succeq D$ would be impossible.) Then A has to be of the form $s \equiv' u$ with $s \succ u$. By (2) we may infer that $C' = D$ hence $t = u$, $\equiv = \equiv'$ and if $U = \sim$ then T or V must be $\sim$. Consider the clause $C''' = D'' \vee \neg(u\,T\,v') \vee s\,U\,v'$ which can be obtained by equality factoring from D. As N is saturated up to redundancy and D is non-redundant, C''' follows from instances of N strictly smaller than D, hence C''' must be true in I_D. D'' and $\neg(u\,T\,v')$ are, however, false in I_D, hence $s\,U\,v'$ is true in I_D so that D cannot be productive.

The other case in which C' produces an inequation $s\,S\,t$ is proved by applying composition resolution as in the proof of the corresponding statement (2) in Lemma 7. In that case, $s\,S\,t$ cannot participate in a rewrite proof for an equation $s\,U\,v'$ in D', hence composition resolution can be restriced to inequations as stated. Details are left to the reader.

(4) and (5) are proved similar to the corresponding statements in Lemma 7. $\Box$

As an immediate corollary of the above lemma we obtain the following completeness theorem.

Theorem 4 *If a set of clauses N is saturated up to redundancy with respect to S_L, then the set $N \cup \mathsf{L}$ is unsatisfiable if and only if it contains the empty clause.*

5.6 Superposition

When restricted to equality $\approx$ as the only binary relation, the subterm chaining calculus specializes to the superposition calculus which is a restricted form of paramodulation. Ordered subterm chaining combines superposition with similar chaining inferences for other binary relations. Paramodulation was introduced by Robinson and Wos (1969), but all early completeness proofs assumed the presence of "functional-reflexive axioms" and required paramodulation into variables (i.e., chaining into variables). Brand (1975) was the first to prove that the functional-reflexive axioms are not needed, and his proof requires only a very limited form of paramodulation into a variable. The first proof that paramodulation into a variable is not needed at all was given by Peterson (1983), while Hsiang and Rusinowitch (1991) were the first to explicitly put some ordering restrictions on paramodulation. The more restrictive superposition calculus described in (Zhang and Kapur 1988, Zhang 1988) turned out to be incomplete; for a counterexample see Bachmair and Ganzinger (1990). The superposition calculus with equality factoring was introduced by Bachmair and Ganzinger (1990). There are also alternatives to equality factoring: Rusinowitch (1991) weakens some ordering constraints, while merging paramodulation is proposed in (Bachmair and Ganzinger 1990), see also (Pais and Peterson 1991). More recently new paramodulation and superposition calculi have been proposed by Bachmair, Ganzinger, Lynch and Snyder (1995b) and by Nieuwenhuis and Rubio (1992). Superposition has also been extended to support associativity and commutativity of functions symbols by Bachmair and Ganzinger (1995a), Nieuwenhuis and Rubio (1994), and Vigneron (1994).

5.7 Monotonicity and Anti-Monotonicity

The completeness results for the subterm chaining calculus are based on an application of standard rewriting techniques for treating replacement axioms in an ordered fashion. They apply to partial congruences. In practice, however, one often encounters theories with non-symmetric relations that satisfy monotonicity properties. For example, to define ordered ring structures we need axioms such as

$$\begin{array}{rcl} x < y & \rightarrow & x + z < y + z \\ x < y & \rightarrow & -y < -x \end{array}$$

that express monotonicity and anti-monotonicity, respectively, of certain functions with the (non-symmetric) ordering relation. Naturally, (subterm) chaining methods for general clauses (Manna and Waldinger 1986, Manna and Waldinger 1992) and completion-like procedures for unit clauses (Levy and Agustí 1993) have been proposed for such relations, but for completeness chaining into some of the variables of the functional-reflexive axioms (cf. the "variable instance pairs" in Bachmair, Dershowitz and Hsiang 1986) is required, which is impractical in general.

For example, given two inequalities $x < x * x$ and $a < b$, and a syntactic ordering in which $a \succ b$ and $x * x \succ x$, we need the "variable subterm chaining" (or "variable overlap") $a < a * a < a * b$ to derive $a < a * b$.[9] If $<$ were symmetric, we would also have a rewrite proof

$$a \Rightarrow^{<} b \Leftarrow^{<} b * b \Leftarrow^{<} a * b$$

for $a < a * b$, and the chaining would be unnecessary. This example, which is typical of the problems variable overlaps may cause for completion procedures, depends on the syntactic ordering $\succ$ and on the non-linearity of the variable x.

In the case of non-unit clauses, proper handling of negative literals poses further difficulties. Take the set of two clauses

$$\begin{array}{rcl} & \rightarrow & a < b \\ f(x) < f(f(y)) & \rightarrow & \end{array}$$

(and the monotonicity axiom $x < y \rightarrow f(x) < f(y)$). This set of clauses is unsatisfiable: from the first clause and the monotonicity axiom we get $f(f(a)) < f(f(b))$, which contradicts the second clause. However, we cannot get a contradiction from the first two clauses by chaining even at variable subterms (regardless of how the ordering is defined). In this example we can get a contradiction by an inference that combines chaining into a variable with a subsequent unification of terms. Thus, we get $f(x) < f(f(a))$ by chaining, and the empty clause, because $f(x)$ and $f(f(a))$ are unifiable. This is a limited form of resolution, which may be called *context resolution*, in which a context is put around one of the literals before the actual resolution takes place. In the example above, the context is determined by the chaining and the subsequent unification. Unfortunately, in the presence of non-linear variables, the context may have to be guessed. Take the clauses $a < b$ and $f(g(b), x, x) \not< f(y, y, g(a))$. Assuming suitable monotonicity axioms, the first clause implies $f(g(b), g(a), g(a)) < f(g(b), g(b), g(a))$, which indicates the appropriate context for $a < b$, so that resolution with the second clauses yields a contradiction.

[9]At this point the reader should not be confusing the ordering $<$ on the object level with the term orderings $\succ$ on the meta-level that is used to restrict the search space for theorems about $<$.

It appears as if the problem cannot be solved satisfactorily in general. Monotonicity axioms cannot be implemented by subterm chaining without avoiding chaining below variables. The alternative which we can propose on the basis of our results is to make monotonicity axioms for general transitive relations subject to standard ordered (head) chaining, but at the same time apply a more refined analysis of redundancy to especially prolific kinds of inferences. For instance, ordered chaining inferences in B of the form

$$\frac{x < y \rightarrow f(x) < f(y) \quad y < z \rightarrow f(y) < f(z)}{x < y\,,\, y < z \rightarrow f(x) < f(z)}$$

with $<$ transitive, are redundant in any N that contains $x < y \rightarrow f(x) < f(y)$. For the inference to be ordered, one must have $y \succ x$ and $y \succ z$. To show redundancy we have to find a proof of the conclusion that uses only instances of N which are smaller than the maximal premise $y < z \rightarrow f(y) < f(z)$. In addition the proof may use arbitrary instances of the transitivity axiom for $<$. Clearly, $x < y \wedge y < z$ implies $x < z$ by transitivity and any instance of $x < z \rightarrow f(x) < f(z)$ is smaller than the corresponding instance of $y < z \rightarrow f(y) < f(z)$ whenever $y \succ x$.[10]

In the future one should develop special versions of subterm chaining for particular algebraic structures such as ordered groups and fields or lattices. The presence of additional algebraic structure seems to simpifiy the problem considerably. On the other hand the technical treatment gets much more involved. The present treatment has to be generalized to rewriting modulo AC, and specific forms of simplification that are afforded by the algebraic structure have to be integrated. A detailed analysis of redundancy is required for inferences between theory clauses and "user clauses". A specialized version of AC-superposition for commutative rings, but without orderings, has been described by Bachmair, Ganzinger and Stuber (1995a).

6 Total Orderings

In this section we shall refine the subterm chaining calculus S in the presence of total orderings. We assume that L specifies $<$ as a transitive relation. Atoms of the form $s < t$ are called *inequalities*. Negative inequalities $\neg(s < t)$ will also be written as $s \not< t$. We assume that $<$ is intended to denote a total ordering, that is, consider saturation of sets N of clauses which contain the irreflexivity axiom

$$x \not< x$$

subsequently also denoted by Ir, and the totality axiom

$$x < y \vee y < x \vee x \approx y\,,$$

subsequently also denoted by $\mathsf{T}_{x,y}$ or, in short, by T. We will show that saturation can be organized such that no explicit inferences with totality are required and such that inferences with clauses in which the maximal term occurs more than once, in particular composition resolution inferences with inequations, can be drastically simplified. Finally we will again address the problem of variable chaining and present simplification techniques that eliminate the need for many, if not all, variable chainings.

[10]This reasoning requires compatibilty of $\succ$ with contexts. Hence even if one does not implement monotonicity axioms by subterm chaining, $\succ$ should be a reduction ordering so as to allow for redundancy arguments of the indicated kind.

In order to not complicate the presentation technically we assume that we have just one total ordering $<$ and that $<$ engages in no composition laws other than transitivity of $<$ and compatibility with congruences. We assume moreover that $\sim$ and equality $\approx$ are the same and the only congruence relation.

6.1 Treating Irreflexivity

It is useful to treat the irreflexivity of $<$ explicitly by an inference we call irreflexivity resolution:

Irreflexivity Resolution:

$$\frac{C\,,\,u<v}{C\sigma}$$

where (i) σ is the most general unifier of u and v; and (ii) $u\sigma < v\sigma$ is strictly maximal with respect to $C\sigma$.

In the presence of irreflexivity resolution negative chaining into $x \not< x$ is redundant:

Proposition 8 *Let N be such that all inferences by irreflexivity resolution from N are redundant in N. Then any inference by negative chaining into the irreflexivity clase $x \not< x$ is redundant in $N \cup \mathsf{Ir}$.*

Proof. Suppose that

$$\frac{C\,,\,s<t \quad s\not<s}{C\,,\,t\not<s}$$

is a ground inference by negative chaining into the irreflexivity clause such that $s \succ t$. Let N' denote the instances of $N \cup \mathsf{Ir}$ smaller than $s \not< s$. We have to prove that $C \vee t \not< s$ is implied by $s<t$, L and N'. Suppose that C is false and $t<s$ is true in some model J of L, $s<t$, and N'. Then $t<t$ is true in J. But $t \not< t$ is in N' which is a contradiction. Hence $s \not< t$ is true in J. The other type of inference by negative chaining into the irreflexivity clause has the form

$$\frac{C\,,\,s<t \quad s\not<s}{C\,,\,t\not\approx s}.$$

where $s \succeq t$. If $s \succ t$, its redundancy is proved in a similar way. For $s = t$ we may obtain C by irreflexivity resolution, and this inference was assumed to be redundant. □

6.2 Treating Totality

At first we show that chaining inferences with totality are not needed.

Proposition 9 *Any chaining inference from $N \cup \mathsf{T}_{x,y}$ of the form*

$$\frac{C\,,\,s<t \quad x<y\,,\,y<x\,,\,x\approx y}{C\,,\,s<y\,,\,y<t\,,\,t\approx y}$$

is redundant in $N \cup \mathsf{T}_{x,y}$.

Proof. We have to show that any ground instance of the conclusion $C' = C\,,\, s < y\,,$ $y < t\,,\, t \approx y$ follows from $s < t$ and from instances in $N \cup \mathsf{T}_{x,y}$ strictly smaller than $\mathsf{T}_{t,y}$, assuming the axioms in L to be satisfied. From the ordering restrictions of the inference we infer that $t \succ s$ and $t \succ y$ for the ground instance under investigation. Suppose that C is false. We perform a case analysis on $s < y$, hence apply $\mathsf{T}_{s,y}$ which is smaller than $\mathsf{T}_{t,y}$. If $s < y$ is true then C' is true. If $y < s$ or $y \approx s$ is true, by transitivity of $<$ and by compatibility with $\approx$, $y < t$ is true, hence C' is true, also in this case. This completes the redundancy proof. □

Chaining inferences of the form

$$\frac{C\,,\, s \approx t \quad x < y\,,\, y < x\,,\, x \approx y}{C\,,\, s < y\,,\, y < t\,,\, t \approx y}$$

into a variable x are excluded anyway in S. Hence if the only composition laws in which $<$ is involved are transitivity and compatibility with $\approx$, no chaining inferences which involve totality are needed.

Lemma 8 *For any set of clauses N, all inferences from $N \cup \mathsf{T}$ by ordered subterm chaining, ordered factoring and irreflexivity resolution in which at least one of the premises is T are redundant in $N \cup \mathsf{T}$.*

Proof. The chaining cases have been considered above. For the other inferences an easy inspection reveals that from T they produce tautologies. □

Next we look at negative chaining into inequations. These can be avoided by transforming clauses of the form $C \vee s \not< t$ into $C \vee t < s \vee t \approx s$ which, in the presence of totality, is logically equivalent.

Proposition 10 *$C \vee s \not< t$ is not an instance of the irreflexivity clause then it is redundant in $N \cup \mathsf{Ir} \cup \{\, C \vee t < s \vee t \approx s \,\}$.*

Proof. For the proof we may assume that $C \vee s \not< t$ is ground, and otherwise apply the proof to all ground instances of $C \vee s \not< t$. If $s = t$ then C is not the empty clause and hence is properly subsumed by the irreflexivity clause. Otherwise, suppose that $s \succ t$ (the case $t \succ s$ is proved in a similar way). Let N' be the set of ground instances of $N \cup \mathsf{Ir} \cup \{\, C \vee t < s \vee t \approx s \,\}$ smaller than $C \vee s \not< t$. $C' = C \vee t < s \vee t \approx s$ is in N'. Let J be a L-model of N'. Suppose that C is false and $s < t$ is true in J. As C' is true in J, $t < s$ or $t \approx s$ is true in J. According to L, $<$ is transitive and compatible with $\approx$. Therefore we conclude in both cases that $t < t$ is true in J which contradicts the fact that $t \not< t$ is in N'. Hence $s \not< t$ is true in J. □

Note that through applying one step of negative chaining with T, followed by reflexivity resolution, from $C \vee s \not< t$ we may deduce $C \vee t < s \vee t \approx s$, if s or t is a maximal term. The significance of the proposition rests in the fact that once we have the latter clause, any inference from the former is redundant. Therefore we need not chain into negative inequations of any clause provided we make totality resolution *mandatory*:

Totality resolution:

$$\frac{C\,,\, u \not< v}{C\,,\, v < u\,,\, v \approx u}$$

where $u \not< v$ is maximal in C.

Inference rules in which the conclusion renders one of its premises redundant are called *simplification rules*. They can be used to *replace* the redundant premise by the conclusion.

The following proposition provides us with further possibilities for optimization.

Proposition 11 *Let C be a ground clause $C' \vee s < t \vee s < u$ with s the maximal term in C and $s \succ t \succ u$. Then C is redundant in*

$$N \cup \mathsf{T} \cup \{\, C' \vee s < u \vee u < t \vee u \approx t,\ C' \vee s < t \vee t < u \,\}.$$

Symmetrically, if C is of the form $C' \vee t < s \vee u < s$ with maximal term s and $s \succ t \succ u$. Then C is redundant in

$$N \cup \mathsf{T} \cup \{\, C' \vee u < s \vee t < u \vee t \approx u,\ C' \vee t < s \vee u < t \,\}.$$

Proof. $C' \vee s < u \vee u < t \vee u \approx t$ and $C' \vee s < t \vee t < u$ are both smaller than C and, in the presence of $\mathsf{T}_{u,t}$ (which is also smaller than C), imply C in any L-model. The symmetric case is essentially the same. □

The proposition motivates the following inferences.

Inequality factoring:

$$\frac{C\,,\, s < t\,,\, s' < u}{C\sigma\,,\, s\sigma < u\sigma\,,\, u\sigma < t\sigma\,,\, u\sigma \approx t\sigma}$$

and

$$\frac{C\,,\, s < t\,,\, s' < u}{C\sigma\,,\, s\sigma < t\sigma\,,\, t\sigma < u\sigma}$$

where σ is the most general unfier of s and s', $t\sigma \not\succeq s\sigma$, and $s\sigma < t\sigma$ is strictly maximal in $C\sigma \vee s\sigma < u\sigma$;

and

$$\frac{C\,,\, t < s\,,\, u < s'}{C\sigma\,,\, u\sigma < s\sigma\,,\, t\sigma < u\sigma\,,\, t\sigma \approx u\sigma}$$

and

$$\frac{C\,,\, t < s\,,\, u < s'}{C\sigma\,,\, t\sigma < s\sigma\,,\, u\sigma < t\sigma}$$

where σ is the most general unfier of s and s', $t\sigma \not\succeq s\sigma$, and $t\sigma < s\sigma$ is strictly maximal in $C\sigma \vee u\sigma < s\sigma$.

6.3 Ordered Chaining for Total Orderings

We are now in a position to define an optimized chaining inference system $\mathsf{O}_{\mathsf{L}}^{\succ}$ for total orderings.

The calculus $\mathsf{O}_{\mathsf{L}}^{\succ}$ consists of the inferences in S_{L} with

(i) the exception of negative chaining or ordered resolution inferences with negative inequations $s \not< t$;

(ii) the additional restriction $s\sigma \neq s'\sigma$ for inferences with a most general unifier σ for which one of the premises takes the form $C = C' \vee s < t \vee s' < u$ [or $C' \vee t < s \vee u < s$], with $s\sigma$ the maximal term in C; and

(iii) the *addition* of inferences by irreflexivity resolution, totality resolution and inequality factoring.

As an immediate consequence of the above investigations we obtain the following completeness theorem.

Theorem 5 *If a set of clauses N is saturated up to redundancy with respect to $\mathsf{O}_{\mathsf{L}}^{\succ}$, then the set $N \cup \mathsf{L} \cup \mathsf{Ir} \cup \mathsf{T}$ is unsatisfiable if and only if it contains the empty clause.*

Proof. We show that if N is saturated up to redundancy by O then $N \cup \mathsf{Ir} \cup \mathsf{T}$ is saturated by S. The inferences of type (i) of S that are excluded from C are redundant. In case they involve the irreflexivity clause this can be seen by applying the Proposition 8. Otherwise if C is a clause $C' \vee s \not< t$ then, as N is saturated by totality resolution, any ground instance $C\sigma$ of C is redundant, or else the ground instance $C''\sigma$ of $C'' = C' \vee t < s \vee t \approx s$ follows from instances of N smaller than $C\sigma$. In the latter case we may apply Proposition 10 to infer that $C\sigma$ follows from instances in $N \cup \{C''\}$ smaller than $C\sigma$. Hence $C\sigma$ is redundant in N.

For inferences of type (ii), let $C\sigma$ be a ground instance of $C = C' \vee s < t \vee s' < u$ [or $C' \vee t < s \vee u < s$] such that $t\sigma \not\preceq s\sigma$, $s\sigma = s'\sigma$, and $s\sigma < t\sigma$ [$t\sigma < s\sigma$] is strictly maximal in C. As N is saturated under inequality factoring, applying Proposition 11, we infer in a similar way that $C\sigma$ is redundant in N.

Finally, from Lemma 8 we conclude that (the remaining) inferences in O that involve the totality axiom are redundant, too. $\square$

6.4 A Refined Clause Ordering

In $\mathsf{O}_{\mathsf{L}}^{\succ}$ composition resolution inferences involving $<$ are still possible. They might, for instance, take the form

$$\frac{C\,,\, u\,V\,v\,,\; u' < v' \quad D\,,\, s < t}{D\sigma\,,\, t\sigma \not< v'\sigma\,,\, s\sigma < v'\sigma}$$

for a relation V different from $<$. Such inferences can be avoided if the ordering on atoms is stratified such that $u\sigma\, V\, v\sigma$ cannot lie in between two inequations involving the maximal term $u\sigma = s\sigma = u'\sigma$.

For this reason the complexity measure c_L for literals L will now be refined. The resulting class of clause orderings $\succ$ will also make $\mathsf{O}^{\succ}$ monotone. Another consequence of the refinement is that most cases of variable elimination for dense total orderings become simplifications, cf. Section 6.6. To that end certain properties of the non-ground clause from which a ground clause is obtained as an instance will be essential. Let L be a ground literal A or $\neg A$ in a ground instance $C\sigma$ of a clause C. Let n_C be the number of different variables occuring as arguments of an equality or an inequality in C. For example, if C is the clause $x < f(y) \vee z \leq f(z)$, then $n_C = 2$, as both x and z (but not y) qualify. Then $c_L = (max_L, p_L, s_L, r_L, n_L, d_L)$ where the first three components are defined as in Section 5.3; r_L is 1 if A is an inequation, r_L is 0 if A is an equation, and r_L is 2, otherwise; d_L is 1 if the maximal term occurs twice in L, and 0, otherwise; and n_L is 0 if A is not an inequation; n_L is -1 if C is the totality axiom, and n_L is n_C, otherwise. Clearly, literal orderings (or more precisely, orderings on occurrences of literals in ground instances of clauses) which are compatible with the new definition of c_L are admissible literal orderings in the sense of section 5.3, thus the completeness results about S and C apply. From now an ordering $\succ$ is called *admissible* if it is compatible with the preceding complexity measure.

Under this class of orderings, any composition resolution involving $<$ must be of the form

$$\frac{C\,,\, u < v\,,\, u' < v' \quad D\,,\, s < t}{D\sigma\,,\, t\sigma \not< v'\sigma\,,\, s\sigma < v'\sigma}$$

where σ is the most general unifier of u, u' and s, $t\sigma \not\succeq s\sigma$, $v\sigma \not\succeq u\sigma$, $v'\sigma \not\succeq u'\sigma$, and the literal $u\sigma < v\sigma$ is strictly maximal in $C\sigma \vee u'\sigma < v'\sigma$. (The inference might also be of the dual form with maximal terms on the right sides of inequations.) Inferences of this form are excluded from $\mathsf{O}^<$ as their first premise is rendered redundant by inequality factoring.

Lemma 9 *For the class of admissible orderings $\succ$, $\mathsf{O}^\succ$ is monotone.*

Proof. Composition resolution inferences do not involve $<$ in their main atom, hence the new component n_L is irrelevant there, and the proof is similar to Lemma 4. Non-trivial cases of inferences are those in which n_C may be bigger for the conclusion than for any of its premises. An easy inspection reveals that this is possible only for chaining inferences through an equality or inequality. As n_C only affects the complexity of inequalities, we need to look at ground inferences of the form

$$\frac{C\,,\, u \approx v \quad D\,,\, s[u] < t}{C\,,\, D\,,\, s[v] < t}$$

or

$$\frac{C\,,\, v < s \quad D\,,\, s < t}{C\,,\, D\,,\, v < t}\,.$$

According to the ordering restrictions in O, in particular (ii) above, neither C nor D contains an inequation of the form $s < w$. Therefore, if L is a literal in C or D, it is smaller than $s < t$ already according to one of the first four components in c_L. (If the complexity measures do not differ with respect to max_L, p_L and s_L, then L cannot be an inequation, hence, r_L ist 0, while it is 1 for $s < t$.) We may conclude that the conclusion is smaller than the second premise, regardless of the number of unshielded variables in it. $\Box$

6.5 Non-strict Orderings

Totality resolution converts $s \not< t$ into a disjunction $t < s \vee t \approx s$, by which the terms s and t are duplicated. In general clauses may get twice as big and chaining produces even longer clauses containing many repeated occurrences of terms. This is a severe efficiency problem in practice. As a solution we propose to deal with $<$, $\approx$ and the reflexive closure $\leq$ of $<$ simultaneously. We restrict ourselves to positive occurrences of strict and non-strict inequations.

Formally we introduce $\leq$ on the meta-level by considering an atom $s \leq t$ just as a more concise notation for the disjunction $s < t \vee s \approx t$. Note that in this disjunction the equation cannot be maximal so that inferences with $\leq$ result from inferences with the strict inequation. We propose the following changes to inferences involving $<$ and $\leq$ which considerably reduce term duplication.

The original chaining inference from the expanded versions of clauses $C \vee s < t$ and $D \vee t \leq u$ yields $C \vee D \vee s < u \vee t \approx u$. We should instead infer $C \vee D \vee s < u$ which is logically sound in the presence of L and which subsumes the former. From $C \vee s \leq t$ and $D \vee t \leq u$, instead of inferring $C \vee D \vee s < u \vee t \approx u \vee s \approx t$, we may deduce $C \vee D \vee s < u \vee t \approx u$ which again is sound and subsumes the former. Given $C \vee s \approx t$ and $D \vee t \leq u$, $\mathsf{S}^{\succ}$ requires to infer $C \vee D \vee s < u \vee t \approx u$. We may instead infer $C \vee D \vee s \leq u$, which stands for $C \vee D \vee s < u \vee s \approx u$. In interpretations in which $s \approx t$ both clauses are equivalent but the latter is smaller than the former whenever $s \prec t$. Hence the old inference becomes redundant whenever the new inference is redundant. Inequality factoring can be changed in a similar way to avoid duplication of terms. An example for the new inferences is

$$\frac{C\,,\, s < t\,,\, s \leq u}{C\,,\, t \leq u\,,\, s < t}$$

together with

$$\frac{C\,,\, s < t\,,\, s \leq u}{C\,,\, u < t\,,\, s \leq u}\,.$$

6.6 Variable Elimination

From now on we use the symbol $\lhd$ to denote strict and non-strict inequalities $<$ and $\leq$. $\unlhd$ will denote $\lhd$ or $\approx$. When we speak of the priority of a relation symbol among $<$, $\leq$, and $\approx$, we take $<$ to have highest and $\approx$ the lowst priority.

A *variable chaining* is a chaining from $C \vee s \lhd_1 t$ and $D \vee u \lhd_2 v$ through $t\sigma = u\sigma$ in which one of the terms t or u is a variable. As we have already observed in section 3.4, the ordering constraints considerably cut down on the number of variable chainings. More specifically, the ordering constraints can only be satisfied if each term t or u is either a non-variable or an *unshielded variable*; that is, a variable that does not occur in a subterm $f(\ldots, x, \ldots)$ nor in an atom that is neither an equation nor an inequation. For example, the variable x is unshielded in $a < x \vee x < b$, if a and b are constants. x and y are both shielded in $x \leq y \vee s(y) < s(x)$. Certain unshielded variables can be eliminated in any total ordering. More occurrences of unshielded variables can be eliminated in total orderings without endpoints. If the ordering is also dense *all* unshielded variables can be eliminated.

The following inference eliminates all unshielded occurences of a variable x from a clause if all occurrences of x are in inequations $s \lhd t$.

Variable Elimination:

$$\frac{C\,,\, u_1 \lhd_1 x, \ldots, u_m \lhd_m x\,,\, x \lhd_1' v_1, \ldots, x \lhd_n' v_n}{C\,,\, \bigvee_{1 \leq i \leq m, 1 \leq j \leq n} u_i \lhd_{i,j} v_j}$$

with $n, m \geq 0$, where x is an unshielded variable not occurring in C, u_i and v_i, and where $\lhd_{i,j}$ is the lowest-priority symbol of $\lhd_i$ and $\lhd'_j$.

This inference rule is sound for total dense orderings without endpoints. It has been used by Bledsoe, Kunen and Shostak (1985) in their chaining calculus. Their calculus does not admit predicates other than $\lhd$. (But even then equality $s \approx t$ is implicitly present in the conjunction of two nonstrict inequalities $s \leq t$ and $t \leq s$. To assure completeness they have to explicitly compute inferences with congruence axioms as specified by the set of clauses

$$\bigvee_{1 \leq i \leq n} x_i < y_i \vee \bigvee_{1 \leq i \leq n} y_i < x_i \vee f(x_1, \ldots, x_n) \leq f(y_1, \ldots, y_n)$$

where f ranges over all function symbols.)

Weaker variable elimination rules can be applied to non-dense orderings. For example, in any total ordering $a < b$ is equivalent to $a < x \vee x < b$. In an ordering without left endpoint a disjunction $C \vee \bigvee_i x \approx u_i \vee \bigvee_j t_j < x$, where x does not occur in C or any term t_i, is equivalent to C. In a total, "discrete" ordering, in which $s(x)$ denotes the "successor" of x, the disjunction $a \leq x \vee x \leq b$ is equivalent to $a \leq b \vee a \approx s(b)$. This can also be generalized to a variable elimination rule. Hines (1994) proposes related elimination rules for a certain fragment of the first-order theory of integers.

Lemma 10 *(i) A clause $C \vee s \approx s$ is redundant with respect to any set of clauses.*

(ii) A clause C' of the form $C \vee \bigvee_i x \unlhd_i u_i$ (or $C \vee \bigvee_i u_i \unlhd_i x$) in which i ranges over a non-empty set of indices and x is an unshielded variable that does not occur in C is L-redundant with respect to any set of clauses that contains C.

(iii) Let C' be a clause $C \vee \bigvee_i u_i \lhd_i x \vee \bigvee_j x \lhd_j v_j$ with an unshielded variable x not occurring in C, u_i, v_i, and let C'' be $C \vee \bigvee_{i,j} u_i \lhd_{i,j} v_j$, where $\lhd_{i,j}$ is the lowest-priority symbol of $\lhd_i$ and $\lhd'_j$. Then C' is either a totality axiom or else C' is L-redundant with respect to any set N that contains C'' and the totality axioms.

Proof. (i) C is a tautology and hence redundant.

(ii) C implies C' and is smaller.

(iii) We sketch the main ideas. Let C' be not a variant of the totality axiom. Take ground instances $D' = D \vee \bigvee_i a_i \lhd_i c \vee \bigvee_j c \lhd_j b_j$ and $D'' = D \vee \bigvee_{i,j} a_i \lhd_{i,j} b_j$ of C' and C'', respectively. Since C'' does not contain the unshielded variable x, but otherwise contains the same terms as C', we have $n_{L'} > n_{L''}$ for any pair of literals L' in D' and L'' in D'', respectively. Consider the maximal literal L'' in D''. Then there is an inequality L' in D' with the same or bigger maximal term occurring in the same argument position. Hence, the complexity tuple associated with L' is bigger than the tuple associated with L'' in one of the first five components. In short, we have $D' \succ D''$.

Let T_a be the set of all disjunctions $c < a_i \vee a_i \leq c$ and let T_b be the set of all disjunctions $c < b_j \vee b_j \leq c$. These are ground instances of the totality axiom. It can be shown that D' is true in every L-model of $T_a \cup \{D''\}$ and in every L-model of $T_b \cup \{D''\}$. Also, if none of the terms a_i or c is maximal in D', then clearly all clauses in T_a are smaller than D'. On the other hand, if some term a_i or c is maximal in D', then that term occurs as the first argument of an inequality in D', which we may use to infer that all clauses in T_b are smaller than D'. (Note that the n_L-count for T_b-literals is -1 which is essential in case c is maximal.) In sum, D' is L-redundant with respect to $\mathsf{T} \cup \{D''\}$. □

The lemma not only indicates that variable elimination for clauses of type (i)–(iii) is simplifying, but that in the presence of the totality axioms the premise in any of the elimination rules is rendered redundant by its conclusion. Like totality resolution, the variable elimination rules are simplification rules: their premises can be replaced by the respective conclusion. In other words, if variable elimination is made mandatory, then no other inference rule needs to be applied to a clause with an unshielded variable, which also means that variable chainings are not needed. The compatibility of eager variable elimination with other chaining systems has been shown by Richter (1984) and Hines (1992).

Variable elimination in the presence of arbitrary equations is somewhat more complicated. The cases (i) and (ii) of Lemma 10 are already applicable to clauses with equations. To eliminate an unshielded variable x from a clause D in the remaining cases we translate all (positive and negative) equalities with x into inequalities, and then apply the above variable elimination rule. More precisely, a clause

$$D = C \vee \bigvee_{1 \le i \le m} x \not\approx u_i \vee \bigvee_{1 \le j \le n} x \approx v_j$$

where x is an unshielded variable, is transformed to the formula

$$C \vee \bigvee_{1 \le i \le m} (x < u_i \vee u_i < x) \vee \bigvee_{1 \le j \le n} (x \le v_j \wedge v_j \le x)$$

in which x does not occur in an equality. (We assume that either the disjunction of the negative equations $x \not\approx u_i$ in D is non-empty or else there are occurrences of x in inequalities of the form $x \lhd s$ *and* $t \lhd x$ in C. (If this is not the case then one of the cases (i) or (ii) of Lemma 10 applies.) The formula can be translated into 2^n clauses, to which the variable elimination rule can be applied. Let us denote by Ve_D the set of all these clauses, with x eliminated. Unfortunately, some ground instances of clauses in Ve_D may be larger than the corresponding ground instance of $D\sigma$ with respect to our clause ordering. Consequently, clauses are not necessarily rendered L-redundant by variable elimination. However, those ground instances of D, in which x is instantiated to the maximal term in the clause, are L-redundant.

Lemma 11 *Let D be a clause with an unshielded variable x of the above form and σ be a substitution such that $x\sigma$ is a maximal term in $D\sigma$. Then $D\sigma$ is redundant with respect to any set N that contains the clauses in Ve_D and the totality axioms.*

To sum up these considerations, a clause with an unshielded variable x is in many but not all cases redundant. In any case inferences involving this variable (chaining through x, superposition from x, [in]equality factoring applied to [in]equalities with x as maximal term) are redundant. (The ordering constraints for such inferences require x to be instantiated with the maximal term, in which case the corresponding instances of the C are redundant, as indicated by the lemma.) The lemma thus indicates that variable chaining is not needed if all variable elimination inferences are sound and are made mandatory. This is the case for dense total orderings without endpoints.

Let $\mathsf{Do}(d, l, r)$ denote the set of all clauses

$$x < d(x, y) \, , \, y \le x$$
$$d(x, y) < y \, , \, y \le x$$
$$l(x) < x$$
$$x < r(x)$$

which encodes the properties of a dense ordering without endpoints. Let $\mathsf{CV}^{\succ}$ denote the calculus consisting of the variable elimination inferences and of the inferences in $\mathsf{O}^{\succ}$ excluding chaining inferences through variables as well as any other inference in which an unshielded variable denotes, according to the ordering restrictions of the inference, the maximal term. $\mathsf{CV}^{\succ}$ is refutationally complete (and sound in the presence of $\mathsf{Do}(d, l, r)$). Explicit inferences with the clauses in $\mathsf{Do}(d, l, r)$ are not required.

Theorem 6 *Let N be a set of clauses that is saturated up to* L*-redundancy with respect to* $\mathsf{CV}^{\succ}$*. Then $N \cup \mathsf{Ir} \cup \mathsf{T} \cup \mathsf{Do}(d, l, r)$ is also saturated up to* L*-redundancy, where d, l and r are function symbols not in N.*

Proof. The only possible inferences between clauses in $N \cup \mathsf{T}$ and $\mathsf{Do}(d, l, r)$ would be variable chainings, of which there are none in CV. The only other inference from $\mathsf{Do}(d, l, r)$ which is the chaining of the two axioms for d produces a tautology and is therefore redundant too. Since N is assumed to be saturated, inferences from $N \cup \mathsf{T}$ are also redundant in N and hence in $N \cup \mathsf{T} \cup \mathsf{Do}(d, l, r)$. In short, $N \cup \mathsf{T} \cup \mathsf{Do}(d, l, r)$ is saturated. □

7 Conclusions

We have shown how to extend standard term rewriting techniques in saturation-based theorem proving to binary relations other than equality. Ordered paramodulation has been generalized to ordered (subterm) chaining. By ordered chaining we may implement a large class of composition laws for binary, in general non-symmetric relations. In the paper we have in particular investigated transitive relations (system B), families of general composition laws (system C), composition laws for binary relations in the presence of partial congruences (system S), and total orderings (systems O and CV).

The main advances over previous work on chaining-based inference systems are the following. We have justified the ordering restrictions for chaining inferences by the concept of commutation for rewrite systems. Our calculi can treat more than one special binary relation at the same time Our completeness results apply to arbitrary signatures of non-logical symbols. We have provided for an integrated treatment of equality and other transitive relations whereby known results about superposition are obtained as special cases. The completeness results for the various inference systems have been obtained in the presence of a general notion of redundancy. The compatibility of simplification and deletion techniques can be justified by simply showing that deleted clauses or ignored inferences are redundant. In particular we have given a justification of variable elimination rules for total orderings by showing the redundancy of clauses with unshielded variables. We have also analysed the problem of chaining at or below variables to some extent.

In this paper we have not described any details about how saturation of a clause set up to redundancy might be done in practice. We refer the reader to our papers Bachmair and Ganzinger (1994b) and Bachmair and Ganzinger (1995b) for a detailed exposition of fairness issues and techniques for simplification and redundancy detection which readily carry over to the framework of the present paper.

The theory described here has been implemented in the Saturate system (Ganzinger 1994) and very promising experimental results have been obtained. Exploiting the structure of a background theory does help to find theorems faster. Our prover will do better for theories of transitive relations that satisfy additional logical properties such as symmetry or totality. By contrast automated theorem provers based on traditional approaches in automated theorem proving usually become tied up in exploring large uninteresting parts of the search space when one adds syntactically cumbersome axioms such as transitivity or totality axioms for a relation.

Search should be replaced as much as possible by don't-care non-deterministic simplification, and the latter is intrisically theory-specific. The principle of replacing search by simplification and the problem of integrating mathematical decision procedures has been fundamental in the design of heuristic theorem provers such as the Boyer-More theorem prover or the PVS-System (Owre, Rushby and Shankar 1992). (In particular Boyer and Moore (1988) provide an interesting discussion about how to integrate mathematical theories into a heuristic prover.) The approach taken here, and as exemplified for the case of quantifier elimination (variable elimination) for dense total orderings, sets out for an integration that guarantees refutational completeness in the presence of arbitrary additional vocabulary. The decidable subcases are decided while in the general case search is to a large extent replaced by simplification. Many of the more prolific inferences can shown to be redundant.

With the methods developed in this paper only a first and small step has been taken towards making saturation-based theorem provers know about mathematics. Much work remains to be done. The next steps should be to investigate theorem proving for theories in which transitive relations, in particular orderings, interact with algebraic structure, that is, consider specializations of the calculi of the present papers in the presence of the axioms of an ordered group of field, or of fragments of lattice theory.

References

L. Bachmair and N. Dershowitz, 1986. Commutation, transformation, and termination. In J. H. Siekmann (ed.), *Proc. 8th Int. Conf. on Automated Deduction*, Lecture Notes in Computer Science 230, pp. 5–20. Springer-Verlag, Berlin.

L. Bachmair and H. Ganzinger, 1990. On restrictions of ordered paramodulation with simplification. In M. Stickel (ed.), *Proc. 10th Int. Conf. on Automated Deduction, Kaiserslautern*, Lecture Notes in Computer Science 449, pp. 427–441. Springer-Verlag, Berlin.

L. Bachmair and H. Ganzinger, 1994a. Ordered chaining for total orderings. *Proc. 12th International Conference on Automated Deduction*, Lecture Notes in Artificial Intelligence 814, pp. 435–450. Springer-Verlag, Berlin.

L. Bachmair and H. Ganzinger, 1994b. Rewrite-based equational theorem proving with selection and simplification. *Journal of Logic and Computation* **4**(3): 217–247. Revised version of Technical Report MPI-I-91-208, 1991.

L. Bachmair and H. Ganzinger, 1994c. Rewrite techniques for transitive relations. *Proc. 9th IEEE Symposium on Logic in Computer Science*, pp. 384–393. IEEE Computer Society Press.

L. Bachmair and H. Ganzinger, 1995a. Associative-commutative superposition. In N. Dershowitz (ed.), *Proc. 3rd Int'l Workshop on Conditional and Typed Rewriting,* Jerusalem, Lecture Notes in Computer Science 968, pp. 1–14. Springer-Verlag, Berlin. Revised version of TR MPI-I-93-267, 1993.

L. Bachmair and H. Ganzinger, 1995b. The theory of resolution. In preparation.

L. Bachmair, N. Dershowitz and J. Hsiang, 1986. Orderings for equational proofs. *Proc. Symp. Logic in Computer Science*, pp. 346–357. Boston, Massachusetts.

L. Bachmair, N. Dershowitz and D. Plaisted, 1989. Completion without failure. In H. Ait-Kaci and M. Nivat (eds), *Resolution of Equations in Algebraic Structures, vol. 2*, pp. 1–30. Academic Press.

L. Bachmair, H. Ganzinger and J. Stuber, 1995a. Combining algebra and universal algebra in first-order theorem proving: The case of commutative rings. In E. Astesiano, G. Reggio and A. Tarlecki (eds), *Recent Trends in Data Type Specification*, Lecture Notes in Computer Science 906, pp. 1–29. Springer-Verlag, Berlin.

L. Bachmair, H. Ganzinger, C. Lynch and W. Snyder, 1995b. Basic paramodulation. *Information and Computation* **121**(2): 172–192. Revised version of TR MPI-I-93-236, 1993.

F. Bellegarde and P. Lescanne, 1987. Transformation ordering. In H. Ehrig, R. Kowalski, G. Levi and U. Montanari (eds), *TAPSOFT'87*, Vol. 249, pp. 69–80.

W. Bledsoe, K. Kunen and R. Shostak, 1985. Completeness results for inequality provers. *Artificial Intelligence* **27**: 255–288.

R. S. Boyer and J. S. Moore, 1988. Integrating decision procedures into heuristic theorem provers: A case study of linear arithmetic. In J. E. Hayes, D. Michie and J. Richards (eds), *Machine Intelligence 11*, chapter 5, pp. 83–124. Clarendon Press, Oxford.

D. Brand, 1975. Proving theorems with the modification method. *SIAM Journal of Computing* **4**: 412–430.

N. Dershowitz and J.-P. Jouannaud, 1990. Rewrite systems. In J. van Leeuwen (ed.), *Handbook of Theoretical Computer Science B: Formal Methods and Semantics*, chapter 6, pp. 243–309. North-Holland, Amsterdam.

N. Dershowitz, 1987. Termination of rewriting. *Journal of Symbolic Computation* **3**: 69–116.

N. Dershowitz, 1991. A maximal-literal unit strategy for horn clauses. *Proc. 2nd Int. Workshop on Conditional and Typed Rewriting Systems*, Lecture Notes in Computer Science 516, pp. 14–25. Springer-Verlag, Berlin.

H. Ganzinger, 1991. A completion procedure for conditional equations. *Journal of Symbolic Computation* **11**: 51–81.

H. Ganzinger, 1994. The Saturate system. Available on the world-wide web under URL http://www.mpi-sb.mpg.de/SATURATE/Saturate.html.

A. Geser, 1991. Relative Termination. Ulmer Informatik–Berichte 91-03, Universiät Ulm, Fakultät für Informatik, Ulm, FRG.

L. M. Hines, 1992. Completeness of a prover for dense linear logics. *Journal of Automated Reasoning* **8**: 45–75.

L. M. Hines, 1994. Str$\dot{+}$ve and integers. In A. Bundy (ed.), *Twelfth International Conference on Automated Deduction*, LNAI 814, pp. 416–430. Springer-Verlag, Nancy, France.

J. Hsiang and M. Rusinowitch, 1991. Proving refutational completeness of theorem proving strategies: The transfinite semantic tree method. *Journal of the ACM* **38**(3): 559–587.

J. W. Klop, 1992. *Term Rewriting Systems*, Vol. 2, pp. 2–116. Clarendon Press, Oxford.

J. Levy and J. Agustí, 1993. Bi-rewriting, a term rewriting technique for monotonic order relations. In C. Kirchner (ed.), *Rewriting Techniques and Applications*, Lecture Notes in Computer Science 690, pp. 17–31. Springer, Berlin.

Z. Manna and R. Waldinger, 1986. Special relations in automated deduction. *Journal of the ACM* **33**: 1–60.

Z. Manna and R. Waldinger, 1992. The special-relation rules are incomplete. In D. Kapur (ed.), *11th International Conference on Automated Deduction*, pp. 492–506. Springer LNAI 607, Saratoga Springs, NY.

R. Nieuwenhuis and A. Rubio, 1992. Basic superposition is complete. *ESOP'92*, Lecture Notes in Computer Science 582, pp. 371–389. Springer-Verlag, Berlin.

R. Nieuwenhuis and A. Rubio, 1994. AC-superposition with constraints: No AC-unifiers needed. *Proc. 12th International Conference on Automated Deduction*, Lecture Notes in Computer Science 814, pp. 545–559. Springer-Verlag, Berlin.

S. Owre, J. M. Rushby and N. Shankar, 1992. PVS: A prototype verification system. *11th Int. Conf. on Automated Deduction*, Saratoga Springs, NY, LNCS 607, pp. 748–752. Springer.

J. Pais and G. Peterson, 1991. Using forcing to prove completeness of resolution and paramodulation. *Journal of Symbolic Computation* **11**: 3–19.

G. E. Peterson, 1983. A technique for establishing completeness results in theorem proving with equality. *SIAM J. on Computing* **12**(1): 82–100.

M. Richter, 1984. Some reordering properties for inequality proof trees. Lecture Notes in Computer Science 171, pp. 183–197. Springer-Verlag.

G. Robinson and L. T. Wos, 1969. Paramodulation and theorem proving in first order theories with equality. In B. Meltzer and D. Michie (eds), *Machine Intelligence 4*, pp. 133–150. American Elsevier, New York.

M. Rusinowitch, 1991. Theorem proving with resolution and superposition: An extension of the Knuth and Bendix completion procedure as a complete set of inference rules. *J. Symbolic Computation* **11**: 21–49.

D. S. Scott, 1979. Identity and existence in intuitionistic logic. *Applications of Sheaves, Proc. Durham*, Lecture Notes in Mathematics 753, pp. 660–695.

J. R. Slagle, 1972. Automatic theorem proving for theories with built-in theories including equality, partial orderings, and sets. *Journal of the ACM* **19**: 120–135.

L. Vigneron, 1994. Associative-commutative dedution with constraints. *Proc. 12th International Conference on Automated Deduction*, Lecture Notes in Computer Science 814, pp. 530–544. Springer-Verlag, Berlin.

H. Zhang and D. Kapur, 1988. First-order theorem proving using conditional rewrite rules. In E. Lusk and R. Overbeek (eds), *Proc. 9th Int. Conf. on Automated Deduction*, Lecture Notes in Computer Science 310, pp. 1–20. Springer-Verlag, Berlin.

H. Zhang, 1988. *Reduction, superposition and induction: Automated reasoning in an equational logic*, PhD thesis, Rensselaer Polytechnic Institute, Schenectady, New York.