Ordered Chaining for Total Orderings*

Leo Bachmair[1] and Harald Ganzinger[2]

[1] Department of Computer Science, SUNY at Stony Brook,
Stony Brook, NY 11794, U.S.A,
`leo@sbcs.sunysb.edu`
[2] Max-Planck-Institut für Informatik,
Im Stadtwald, D-66123 Saarbrücken, Germany,
`hg@mpi-sb.mpg.de`

Abstract. We design new inference systems for total orderings by applying rewrite techniques to chaining calculi. Equality relations may either be specified axiomatically or built into the deductive calculus via paramodulation or superposition. We demonstrate that our inference systems are compatible with a concept of (global) redundancy for clauses and inferences that covers such widely used simplification techniques as tautology deletion, subsumption, and demodulation. A key to the practicality of chaining techniques is the extent to which so-called variable chainings can be restricted. Syntactic ordering restrictions on terms and the rewrite techniques which account for their completeness considerably restrict variable chaining. We show that variable elimination is an admissible simplification technique within our redundancy framework, and that consequently for dense total orderings without endpoints no variable chaining is needed at all.

1 Introduction

The axioms of the theories of partial and total orderings are extremely prolific in the context of resolution-based theorem proving. Many theorem provers build in transitivity by so-called chaining rules, which allow one to derive $(C \vee D \vee u < v)\sigma$ from premises $C \vee u < s$ and $D \vee t < v$ after unifying s and t by σ. Even though chaining is a considerable improvement over naive resolution, in this general form it still generates a huge search space. First, chaining inferences are always possible if s or t is a variable. For instance, the totality axiom $x < y \vee y \leq x$ can always be applied in four different ways, leading to the derivation of many equivalent variants of each clause. The situation is even worse for total orderings with additional structure, and in particular dense orderings with no endpoints. The clauses which express density and the absence of endpoints

* The research described in this paper 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.

can also be chained with any other clause. For example, the clause $x < rx$ expressing the non-existence of a right endpoint, generates infinitely many inequalities $x < r^n x$ just by chaining with itself. Secondly, the same clause $C \vee u < s$ can be used as the first or as the second premise of a chaining inference, so that one needs to chain with both terms, u and s. Third, and related to the first two points, chaining is designed to generate the full (possibly infinite) transitive closure of a given set of inequalities, even though for any particular refutation a finite subset will be sufficient. This lack of goal-orientedness forms another obstacle to any efficient proof search. These problems apply to transitive relations in general; solutions have been proposed for certain theories.

Bledsoe and Hines (1980) have investigated dense total orderings without endpoints and have developed techniques for eliminating certain occurrences of variables from formulas. In their inference system, no chaining through variables is performed and no explicit inferences with transitivity, density, totality and the "no endpoints" axioms are computed. Completeness results for particular such systems of restricted chaining are proved by Bledsoe, Kunen and Shostak (1985) and Hines (1992). Theorem provers developed from these theoretical investigations have performed successfully in proving theorems such as the continuity of the sum of two continuous functions or the intermediate value theorem; see Bledsoe and Hines (1980), Hines (1988) or Hines (1990).

Ordered paramodulation (Hsiang and Rusinowitch 1991) and superposition (Bachmair and Ganzinger 1990) are chaining-based inference systems for equality relations (the congruence properties of which also require chaining through subterms). Paramodulation into or below variables was first shown to be unnecessary by Brand (1975). Various syntactic restrictions avoid that equalities are always applied in both directions. In many cases even an infinite transitive closure of a given set of equalities can be represented by a finite convergent rewrite system. Only that system, and not the complete transitive closure of the given set of equalities (or clauses), is computed by the ordered variants of paramodulation and superposition.

Our aim is to combine the two approaches, keeping their best features, but avoiding their drawbacks. For that purpose we adapt the term rewriting techniques for arbitrary transitive relations described in Bachmair and Ganzinger (1993b) to total orderings.

1.1 Results

We present refutationally complete inference systems for total orderings, in which chaining is restricted by syntactic ordering constraints in much the same way as we know it from superposition calculi. That is, chaining inferences as above are performed only if the term $s\sigma$ is maximal in both (instances by σ of the) parent clauses. These ordering constraints immediately rule out many forms of chaining through variables. For instance, a term $x\sigma$ cannot be maximal if the variable x is shielded in a clause, that is, occurs as an argument of a function symbol. We also go beyond simple chaining in that we eliminate all occurrences of the maximal term in one single inference that combines several chaining steps. The effect is similar to hyper-resolution, in that the results of intermediate chaining steps need not be explicitly generated. Explicit inferences with transitivity and totality are shown to be redundant. This inference system applies to arbitrary total orderings and avoids most, though not all, variable chainings. The ordering constraints, like in the equational case, derive from a presentation of the full transitive closure of the given binary relation by an appropriate rewrite closure.

We prove refutational completeness of our inference systems in the presence of a general notion of redundancy for clauses and inferences by which most of the commonly applied simplification and elimination techniques (e.g., tautology elimination or subsumption) can be justified. For total orderings in particular we show that variable elimination, as proposed by Bledsoe and Hines (1980), is a simplification rule in our sense: a clause becomes redundant once variable elimination has been applied to it. In other words, variable elimination can be made mandatory. Ordering constraints for inferences and mandatory elimination of unshielded variables together achieve that chaining through variables can be completely avoided for dense total orderings with no endpoints. Variable elimination is also excluded in the inference systems proposed by Bledsoe, Kunen and Shostak (1985) and Hines (1992), though these calculi, unlike ours, are not compatible with tautology deletion. Our proofs are comparatively simple and in particular profit from our ability of treating simplification techniques such as variable elimination as a separate issue.

The first of the two calculi which we present in this paper lacks an efficient treatment of equality. Like in previous approaches (Bledsoe and Hines 1980, Bledsoe, Kunen and Shostak 1985, Hines 1992) an equality $s \approx t$ is represented by two inequalities $s \leq t$ and $t \leq s$. The disadvantage of this approach is that the implicitly specified equality relation requires additional congruence axioms for each function symbol, while such powerful simplification mechanisms as demodulation cannot be used. Therefore we introduce another system, in which we combine chaining with superposition (Bachmair and Ganzinger 1990). Equality is thus built into the inference rules; explicit inferences with congruence axioms or functional reflexive axioms and superposition into or below variables are not needed. The extended inference system also considerably improves earlier chaining systems with paramodulation (Slagle 1972), and for dense orderings without endpoints superposition from variables is not needed either. In addition, when superposing into inequalities only the maximal term of the inequality needs to be replaced. As before, the completeness result allows one to discard redundant clauses and inferences and admits simplifications such as demodulation or condensement. Certain technical definitions and proofs that had to be omitted due to lack of space, can be found in (Bachmair and Ganzinger 1993a).

We have implemented most the techniques investigated here as an extension of the Saturate system (Nivela and Nieuwenhuis 1993), and have obtained promising experimental results.

2 Preliminaries

2.1 Orderings

A (strict) *partial ordering* is a transitive and irreflexive binary relation; a *quasi-ordering* a reflexive and transitive binary relation. The reflexive closure of a strict ordering is a quasi-ordering. On the other hand, if $\leq$ is a quasi-ordering, then its *strict part* $<$, defined by: $x < y$ if and only if $x \leq y$ but not $y \leq x$, is a strict ordering. An ordering $<$ is said to be *total* if $x < y$ or $y < x$, whenever x and y are distinct. The ordering is *dense* if for all x and y with $x < y$, there exists an element z, such that $x < z$ and $z < y$. An ordering $<$ has no *left* (resp. *right*) *endpoint* if for every x there exists a y such that $y < x$ (resp. $x < y$). By an ordering *without endpoints* we mean one that has neither a left nor a right endpoint. For instance, the usual less-than relation on the natural numbers is a total ordering with a left, but no right, endpoint. The less-than relation on the real numbers is a dense total ordering without endpoints.

2.2 Predicate logic

We consider first-order predicate logic with equality; more specifically, first-order languages with (uninterpreted) function symbols, variables, and the (interpreted) predicate symbols $<$, $\leq$, and $\approx$.[3] 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 $s \approx t$, called an *equality*, or an expression $s < t$ or $s \leq t$, called a strict or non-strict *inequality*, respectively, where s and t are terms. A *literal* is an expression A (a *positive* literal) or $\neg A$ (a *negative* literal), where A is an atomic formula. We also write $s \not\approx t$ instead of $\neg(s \approx t)$ and use a similar notation for inequalities. When we speak of the *priority* of a predicate symbol, we take $<$ to have the highest and $\approx$ the lowest priority. The symbol $\lhd$ is used to denote (strict or non-strict) inequalities; the symbol $\unlhd$ to denote equalities and inequalities. 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.

We intend $\approx$ to be interpreted as an equality relation[4] and $<$ as a strict ordering. Also, $x \leq y$ is meant to be interpreted as $x < y \vee x \approx y$. (From a logical point of view, the symbol $\leq$ is therefore superfluous. But for theorem proving purposes shorter formulas are generally preferable, and it is better to avoid replacing a non-strict inequality by a disjunction.) The most problematic properties of these relations—for an automated theorem prover—are the transitivity properties. By a *transitivity interpretation* we mean a model of the set

[3] Uninterpreted predicate symbols can be encoded in a many-sorted language with function symbols and equality, by representing a formula $P(t_1, \ldots, t_n)$ as an equality $f_P(t_1, \ldots, t_n) \approx true_P$.

[4] An equality relation is a congruence on ground terms.

TR of all *transitivity axioms*

$$x \trianglelefteq_1 y, y \trianglelefteq_2 z \rightarrow x \trianglelefteq z$$

where $\trianglelefteq$ is the highest-priority symbol of $\trianglelefteq_1$ and $\trianglelefteq_2$. For example, $x \not< y \vee y \not\approx z \vee x < z$ is a transitivity axiom. The set PO, consisting of the axioms in TR and the axioms (i) $x \not< x$, (ii) $x \leq x$, and (iii) $x \not< y \vee x \leq y$, encodes that $<$ is a strict ordering and $\leq$ a quasi-ordering (with $<$ contained in its strict part). Interpretations satisfying PO are called partial orderings for short. If in addition the totality axiom (iv) $x < y \vee y \leq x$ is satisfied, the interpretation is called a total ordering. In that case, $<$ coincides with the strict part of $\leq$.

2.3 Chaining

Many theorem provers build in transitivity by so-called *chaining rules*,

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

where σ is a most general unifier of s and t; see Slagle (1972). For instance, with chaining we may deduce $u < v$ from $u < s$ and $s < v$. Specific variants of this inference rule can be found in calculi for dense total orderings (Bledsoe and Hines 1980, Bledsoe, Kunen and Shostak 1985) and in paramodulation calculi for equality (Robinson and Wos 1969). In its full generality, the chaining rule is not practical, as the search space spanned by the inferences may be huge. *Variable chainings*, that is, chainings with a premise $C \vee u < x$ or $D \vee y < v$, are particularly prolific. Fortunately, they can be completely excluded in the case of dense total orderings without endpoints (Bledsoe, Kunen and Shostak 1985) and considerably restrained in the case of paramodulation (Brand 1975 was the first to prove that paramodulation into a variable is unnecessary).

Chaining essentially generates the transitive closure of a given binary relation. For example, from $a < b$ and $b < c$ and $c < d$ we may deduce $a < c$ and $b < d$ and $a < d$. In other words, whenever there is a (finite) chain of equalities and/or inequalities

$$x_1 \trianglelefteq_1 x_2 \trianglelefteq_2 \cdots \trianglelefteq_{n-1} x_n$$

the equality or inequality $x_1 \trianglelefteq x_n$ can be deduced, where $\trianglelefteq$ is the highest-priority predicate among all $\trianglelefteq_i$. The basic idea of the term rewriting approach is to consider not arbitrary chains, but only those in which the intermediate terms $x_2, \ldots, x_{n-1}$ are in a certain sense simpler than the endpoints x_1 and x_n. Moreover, the equality or inequality $x_1 \trianglelefteq x_n$ is not necessarily deduced, but instead the corresponding chain may be implicitly represented by a rewrite system. We employ these ideas for the design of refutationally complete chaining systems in which various constraints are imposed on inference rules. Terminating rewrite systems are important in this context.

2.4 Rewrite systems

Let I be a set of ground atomic formulas and $\succ$ be a simplification ordering. We use the equalities and inequalities in I as *rewrite rules*. More precisely, we write $u \lhd_I v$ if $u \lhd v$ is an inequality in I and write $u \approx_I v$ if $u = w[s]$ and $v = w[t]$, for some term w and equality $s \approx t$ or $t \approx s$ in I. Furthermore, given $u \unlhd_I v$, we write $u \unlhd_I^L v$, if $u \succ v$, and $u \unlhd_I^R v$, if $v \succ u$. The subscripts are dropped if I is clear from the context.

By a *proof* (in I) we mean a finite sequence $u_0 \unlhd_1 u_1 \unlhd_2 u_2 \ldots u_{n-1} \unlhd_n u_n$ (in I), where $n \geq 0$. More specifically, we speak of a proof of $u_0 \leq u_n$. We speak of a proof of $u_0 < u_n$ if in addition at least one of the symbols $\unlhd_i$ is $<$ (and hence $n \geq 1$); and of a proof of $u_0 \approx u_n$ if *all* symbols $\unlhd_i$ are $\approx$. (If $n = 0$, we have a proof of $u_0 \approx u_0$.) By the *transitive closure* of I we mean the set of all atoms $u \unlhd v$ provable in I. A *rewrite proof* is a sequence

$$u_0 < u_1$$

or

$$u_0 \unlhd_1^L \cdots \unlhd_{m-1}^L u_m \unlhd_m^R \cdots \unlhd_n^R u_n.$$

By the *rewrite closure* of I we mean the set of all atoms $u \unlhd v$ that are provable by a rewrite proof in I.

In proving the completeness of chaining systems we construct Herbrand models for certain clause sets, describing interpretations by rewrite closures. A key question is under which circumstances the rewrite closure of a set I is a transitivity interpretation; a question, it turns out, that is related to commutation properties of the rewrite relations $\unlhd_I$.

If a sequence $u_0 \unlhd_1 \cdots \unlhd_n u_n$ is not a rewrite proof then either (i) there is a subsequence $u_{i-1} \unlhd_i^R u_i \unlhd_{i+1}^L u_{i+1}$, or (ii) u_{i-1} and u_i are identical or incomparable (i.e., $u_{i-1} \not\succ u_i$ and $u_i \not\succ u_{i-1}$), for some i. A subsequence $u \unlhd_1^R t \unlhd_2^L v$, called a *peak*, is said to *commute* if $u \unlhd v$ is provable by a rewrite proof, where $\unlhd$ is the highest-priority symbol of $\unlhd_1$ and $\unlhd_2$. Commutation allows us to deal with case (i). If $\succ$ is a complete simplification ordering, then case (ii) applies only if $u_{i-1} = u_i$.

For example, if I contains the atoms $a < b$ and $b < c$ and $c < d$, and if $a \succ b \succ c \succ d$, then $a <^L b$ and $b <^L c$ and $c <^L d$. The relation $<^L$ evidently commutes with the (empty) relation $<^R$. If we choose the ordering differently, $b \succ c \succ d \succ a$, there is a peak, $a <^R b <^L c$, that does not commute. There are rewrite proofs of $a < b$ and $b < c$, but not of $a < c$.

The following lemma relates commutation and transitivity.

Lemma 1. *Let $\succ$ be a complete simplification ordering and I be a Herbrand interpretation that contains no strict inequality $t < t$. The rewrite closure of I is a transitivity interpretation if and only if all peaks in I commute.*

The lemma provides the starting point for our investigation of chaining techniques for total orderings, as peaks can be made to commute by applying suitable chaining inferences. (In the above example the peak $a <^R b <^L c$ commutes once the inequality $a < c$ has been deduced from $a < b$ and $b < c$ by chaining.) Similar, so-called "critical pair lemmas" form the basis of all completion procedures;

Levy and Agustí (1993) appear to have been the first to apply these techniques to non-symmetric rewrite relations. We go beyond usual completion procedures in that we consider general clauses, and thus have to deal with negative literals and disjunctions of literals. Particular emphasis will be given to the question of (the necessity of) variable chainings. For a discussion of the general aspects of these questions we refer to our work on rewrite techniques for transitive relations (Bachmair and Ganzinger 1993b). In this paper we look at total orderings, primarily dense total orderings without endpoints.

3 Maximal chaining

Our inference system is parameterized by complete reduction orderings $\succ$.[5] Let us assume an arbitrary such ordering to be given.

3.1 Inference rules

The predicate $\leq$ for non-strict inequality can be defined in terms of strict inequality and equality. It is also possible, though, to express equality in terms of inequality, representing $u \approx v$ by (the conjunction of) two inequalities $u \leq v$ and $v \leq u$. This, indeed, is the framework in which previous chaining inference systems have been formalized, the advantage being that no specific inference mechanisms for equality are needed. The necessary properties of the implicit equality relation are specified by the set $EE_{\mathcal{F}}$ 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 in $\mathcal{F}$; see Bledsoe, Kunen and Shostak (1985). The necessity of including these axioms is a disadvantage. Later on, we will build equality into the inference mechanism via paramodulation, thereby obviating any explicit equality axioms. (Another advantage of equational inference systems is that further simplification techniques such as demodulation become available, which are indispensable in actual implementations.)

In this section all atomic formulas are assumed to be (strict or non-strict) inequalities. Totality can be expressed as a clause $x < y \vee y \leq x$ or $x \leq y \vee y \leq x$. By TA we denote the set of these two clauses. They can be used to transform negative inequalities into positive ones: replace $x \not< y$ by $y \leq x$ and $x \not\leq y$ by $y < x$. Thus, we only need to consider disjunctions of positive inequalities.

We build the irreflexivity and transitivity properties directly into the inference mechanism.

Irreflexivity Resolution:

$$\frac{C \,,\, s < t}{C\sigma}$$

where σ is the most general unifier of s and t and $s\sigma$ is a maximal term in $C\sigma$.

[5] A reduction ordering is *complete* if it is total on ground terms. Complete reduction orderings on ground terms are simplification orderings.

Maximal Chaining:

$$\frac{C\ ,\ u_1 \lhd_1 s_1, \ldots, u_m \lhd_m s_m \quad D\ ,\ t_1 \lhd'_1 v_1, \ldots, t_n \lhd'_n v_n}{C\sigma\ ,\ D\sigma\ ,\ \bigvee_{1 \leq i \leq m, 1 \leq j \leq n} u_i\sigma \lhd_{i,j} v_j\sigma}$$

where (i) σ is the most general unifier of $s_1, \ldots, s_m, t_1, \ldots, t_n$ and $\lhd_{i,j}$ is the highest-priority symbol of $\lhd_i$ and $\lhd'_j$, (ii) $u_i\sigma \not\succeq s_1\sigma$, for all $1 \leq i \leq m$, and $u\sigma \not\succeq s_1\sigma$, for all terms u in C, and (iii) $v_i\sigma \not\succeq t_1\sigma$, for all $1 \leq i \leq n$, and $v\sigma \not\succ t_1\sigma$, for all terms v in D, and $t_1\sigma$ occurs in $D\sigma$ only in inequalities $v \lhd t_1\sigma$.

For example, from $a < f(c)$ and $f(x) < a \lor f(x) < b$ we may deduce $a < a \lor a < b$ (assuming $f(c) \succ a \succ b$ in the given ordering). Maximal chaining is designed to eliminate the maximal terms in a clause. The conclusion of a maximal chaining inference can also be obtained by a sequence of $m \times n$ ordinary chaining inferences, interspersed with factoring inferences; but an important difference is that with maximal chaining the intermediate clauses need not be deduced.

Thus, from $a < f(c)$ and $f(x) < a \lor f(x) < b$ we may obtain, first $a < a \lor f(c) < b$, and then $a < a \lor a < b$ by ordinary chaining. The intermediate clause $a < a \lor f(c) < b$ cannot be obtained by maximal chaining. In this regard, the effect of maximal chaining is similar to that of hyper-resolution. This, incidentally, is also a significant difference between maximal chaining and "equivalence factored chaining," as defined by Hines (1992). A single equivalence factored chaining step may also combine several ordinary chaining steps, but intermediate clauses can also be deduced. For instance, both $a < a \lor f(c) < b$, and $a < a \lor a < b$ are equivalence factored chain resolvents.[6]

The inference system $MC^{\succ}$ consisting of irreflexivity resolution and maximal chaining represents our basic inference mechanism for total orderings. (We also assume that the premises of an inference have no variables in common; if necessary, the variables in one premise are renamed.) Below we shall outline further improvements to the calculus for richer structures, such as dense total orderings without endpoints. These improvements ultimately derive from the concept of redundancy (of clauses and inferences) discussed in the next section.

3.2 Redundancy and Refutational Completeness

Simplification and deletion techniques, such as subsumption, tautology deletion, condensement, demodulation, contextual rewriting, etc., have proved to be indispensable for actual implementations of theorem provers. As in our previous work on superposition calculi, we provide a general framework for formalizing and reasoning about such techniques, based on an appropriate notion of redundancy.

Redundancy of inferences and clauses depends on a well-founded ordering on clauses that is obtained by lifting the given ordering term ordering $\succ$ first to literals and then to clauses. The ordering is the key to our refutational completeness proofs and therefore unshielded variables are also taken into account

[6] Another difference is that maximal chaining employs an ordering restriction on terms, whereas equivalence factored chaining imposes a "shielding term restriction."

in its definition. Details on the clause ordering, which we denote by the symbol $\succ$ also, can be found in Bachmair and Ganzinger (1993a).

Given sets of clauses N and S, we say that a ground instance $C\sigma$ of some clause C (which need not be an element of N) is *S-redundant* with respect to N if there exist ground instances $C_1\sigma_1, \ldots, C_k\sigma_k$ of N such that (i) $C\sigma$ is true in every model of $S \cup \{C_1\sigma_1, \ldots, C_k\sigma_k\}$ and (ii) $C\sigma \succ C_j\sigma_j$, for all j with $1 \leq j \leq k$. The clause C is S-redundant if all its ground instances are.

Tautologies are S-redundant in this sense, for any set S, and most cases of proper subsumption are also covered by this notion of redundancy. According to the definition, the axioms in S are all S-redundant. We are particularly interested in PO-redundancy. (Since these axioms are built into the chaining mechanism, we expect them to be redundant.)

A ground inference with conclusion B and maximal premise C is called *S-redundant* with respect to N if either some premise is S-redundant, or else there exist ground instances $C_1, \ldots, C_k$ of N such that B is true in every model of $S \cup \{C_1, \ldots, C_k\}$ and $C \succ^c C_j$, for all j with $1 \leq j \leq k$. A non-ground inference is called S-redundant if all its ground instances are S-redundant.

We say that a set of clauses N is *saturated up to S-redundancy* if all inferences from N are S-redundant. Since the ground versions of the inferences in $MC^{\succ}$ are simplifying in that their conclusion is smaller than their maximal premise, they can be rendered redundant by adding the conclusion to the given set of clauses. Thus, computing the closure of a clause set under these inferences yields a saturated set.

The following theorem establishes the refutational completeness of MC in the presence of redundancy.

Theorem 2. *Let N be a set of clauses with subset $EE_{\mathcal{F}}$, where $\mathcal{F}$ is the given set of function symbols. If N is saturated up to PO-redundancy with respect to MC, then it has a total ordering as model if and only if it does not contain the empty clause.*

This theorem applies to arbitrary total orderings. The chaining mechanism can be improved for more specific theories, such as dense total orderings without endpoints.

3.3 Variable elimination

A *variable chaining* is a maximal chaining in which one of the terms s_i or t_j is a variable. Variable chaining can be quite prolific, as the unification of terms required for an inference always succeeds if the terms are variables. Fortunately, the ordering constraints in conditions (ii) and (iii) considerably cut down on the number of variable chainings. More specifically, the ordering constraints can only be satisfied if each term s_i or t_j is either a non-variable or an *unshielded variable*; that is, a variable that does not occur in a subterm $f(\ldots, x, \ldots)$. For example, the variable x is unshielded in $a < x \vee x < b$. 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.

Variable Elimination:

$$\frac{C\ ,\ u_1 \lhd_1 x, \ldots, u_m \lhd_m x\ ,\ x \lhd'_1 v_1, \ldots, x \lhd'_n v_n}{C\ ,\ \bigvee_{1 \leq i \leq m, 1 \leq j \leq n} u_i \lhd_{i,j} v_j}$$

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. 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 $C \vee \bigvee_i t_i < x$, 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 $a \leq s(b)$. This equivalence may also be used as a variable elimination rule.

Lemma 3. *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 or 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 is PO-redundant with respect to any set N that contains C'' and the totality axioms.*

The lemma not only indicates that variable elimination 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. The variable elimination rule may therefore be called *simplification* rules: their premises can be *replaced* by the respective conclusion. In other words, variable elimination can be made *mandatory*, in that 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 variable elimination with other chaining systems has been shown by Richter (1984) and Hines (1992).

By $CV^{\succ}$ we denote the calculus consisting of irreflexivity resolution and maximal chaining plus variable elimination. Also, let DO denote the set PO plus certain appropriately chosen clauses to encode the properties of a dense ordering without endpoints.

Lemma 4. *Let N be a set of clauses that is saturated up to PO-redundancy with respect to MC and contains $EE_{\mathcal{F}}$ as a subset, where $\mathcal{F}$ is the given set of function symbols. If N contains no unshielded variables, then $N \cup DO$ is also saturated up to PO-redundancy.*

The significance of this lemma rests on the fact that through simplification it is possible to eliminate all unshielded variables from clauses. Clauses $C \vee x \leq x$ are PO-tautologies and, hence, redundant. A clause $C \vee x < x$ becomes redundant once C has been deduced by irreflexivity resolution. (This is the only situation in which irreflexivity resolution needs to be applied to a clause $C \vee s < t$, where s or t is a variable.) All other unshielded variables can be eliminated by variable elimination.

4 Chaining with superposition

The chaining calculus described above requires explicit equality axioms. Equality can also be built in via paramodulation (originally introduced by Robinson and Wos 1969), which in fact may be seen as a form of "subterm chaining" (as discussed in Bachmair and Ganzinger 1993b). We now allow all three symbols $\approx$, $<$ and $\leq$ to occur in clauses. This is logically redundant as $\leq$ can be expressed as a disjunction in $\approx$ and $<$, but from a practical point of view, and as was confirmed by our experimentation with the Saturate system, it is more appropriate to handle the three relations simultaneously and specifically so as to avoid duplication of terms as much as possible. For the same reason we allow negative equalities $s \not\approx t$ that would otherwise have to be transformed into disjunctions $s < t \vee t < s$. Thus, clauses may contain positive and negative equalities in addition to positive inequalities.

Totality TE can be expressed as a clause $x < y \vee y < x \vee x \approx y$ and the anti-symmetry axiom AS as $x \leq y , y \leq x \rightarrow x \approx y$.

The calculus CS consists of three parts: a chaining calculus for inequalities, a superposition calculus for equalities, and further chaining rules that connect equalities with inequalities. Superposition is a form of paramodulation in which ordering constraints are imposed on the inference rules. We use the superposition calculus of Bachmair and Ganzinger (1994). Symmetry of equality can be built in by ordered resolution with the clause $u \not\approx v \vee v \approx u$. (In practice, equations are usually considered as unordered pairs to avoid unnecessary duplication of formulas.)

Reflexivity Resolution:

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

where σ is a most general unifier of u and v and $u\sigma \not\approx v\sigma$ is a maximal literal in $C\sigma$.

Superposition:

$$\frac{C \ , \ t \approx s \quad D \ , \ L'[s']}{C\sigma \ , \ D\sigma \ , \ L[t]\sigma}$$

where (i) σ is a most general unifier of s and s'; (ii) $t\sigma \approx s\sigma$ is strictly maximal in $C\sigma$ and $t\sigma \not\leq s\sigma$; (iii) L' is either an equality $u[s] \approx v$ or $v \approx u[s]$, or the negation thereof, such that $v\sigma \not\leq u\sigma$ and $L'\sigma$ is strictly maximal in $D\sigma$, if it is a positive equality, and just maximal in $D\sigma$ if it is negative; and (v) s' is not a variable.

Equality Factoring:

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

where (i) σ is a most general unifier of s and s', (ii) $t\sigma \not\leq s\sigma$, and (iii) $s\sigma \approx t\sigma$ is maximal in $(C \vee s \approx t \vee s' \approx t')\sigma$.

Superposition is a restricted form of subterm chaining applied to equations. We need a similar form of "maximal subterm chaining" with equations into inequations. These inference rules are designed to reduce maximal terms.

Equality Chaining Left:

$$\frac{C\ ,\ u \approx s \quad D\ ,\ t_1[s_1]_p \lhd_1 v_1, \ldots, t_n[s_n]_p \lhd_n v_n}{C\sigma\ ,\ D\sigma\ ,\ t_1[u]_p \lhd_1 v_1, \ldots, t_n[u]_p \lhd_n v_n}$$

where (i) $\sigma = \rho\tau$ with ρ the most general unifier of $s, s_1, \ldots, s_n$, and τ the most general unifier of $t_1[s_1]\rho, \ldots, t_n[s_n]\rho$, (ii) $u\sigma \not\preceq s\sigma$, and $u\sigma \approx s\sigma$ is strictly maximal in $C\sigma$, (iii) $v_i\sigma \not\succeq t_1\sigma$, for all $1 \leq i \leq n$, and $v\sigma \not\succ t_1\sigma$, for all terms v in D, and $t_1\sigma$ occurs in $D\sigma$ only in equalities or inequalities $v \unlhd t_1\sigma$, and (iv) none of the s_i is a variable.

Equality Chaining Right:

$$\frac{C\ ,\ u \approx s \quad D\ ,\ v_1 \lhd_1 t_1[s_1]_p, \ldots, v_n \lhd_n t_n[s_n]_p}{C\sigma\ ,\ D\sigma\ ,\ v_1 \lhd_1 t_1[u]_p, \ldots, v_n \lhd_n t_n[u]_p}$$

where (i) $\sigma = \rho\tau$ with ρ is the most general unifier of $s, s_1, \ldots, s_n$, τ the most general unifier of $t_1[s_1]\rho, \ldots, t_n[s_n]\rho$, (ii) $u\sigma \not\preceq s\sigma$, and $u\sigma \approx s\sigma$ is strictly maximal in $C\sigma$, (iii) $v_i\sigma \not\succeq t_1\sigma$, for all $1 \leq i \leq n$, and $v\sigma \not\succ t_1\sigma$, for all terms v in D, and $t_1\sigma$ occurs in $D\sigma$ only in equalities, and (iv) none of the s_i is a variable.

Finally, we also need to be able to deduce equalities implicitly specified by inequalities, so that the anti-symmetry axiom is satisfied. For that purpose we modify the maximal chaining rule as follows.

Inequality Chaining:

$$\frac{C\ ,\ u_1 \lhd_1 s_1, \ldots, u_m \lhd_m s_m \quad D\ ,\ t_1 \lhd'_1 v_1, \ldots, t_n \lhd'_n v_n}{C\sigma\ ,\ D\sigma\ ,\ \bigvee_{1 \leq i \leq m, 1 \leq j \leq n} ineq(i,j)}$$

where (i) σ is the most general unifier of $s_1, \ldots, s_m, t_1, \ldots, t_n$, (ii) $ineq(i,j)$ is the strict inequality $u_i\sigma < v_j\sigma$ if $\lhd_i$ or $\lhd'_j$ is $<$, and the disjunction $u_i\sigma < v_j\sigma \vee u_i\sigma \approx t_1\sigma$ otherwise,[7] (iii) $u_i\sigma \not\succeq s_1\sigma$, for all $1 \leq i \leq m$, and $u\sigma \not\succeq s_1\sigma$, for all terms u in C, and (iv) $v_i\sigma \not\succeq t_1\sigma$, for all $1 \leq i \leq n$, and $v\sigma \not\succ t_1\sigma$, for all terms v in D, and $t_1\sigma$ occurs in $D\sigma$ only in inequalities $v \lhd t_1\sigma$.

For example, from $a \leq f(x)$ and $f(a) \leq b \vee f(y) < c$ we may deduce $a < b \vee f(a) \approx a \vee a < c$ (assuming $f(a)$ is the maximal term in the clause). Inequality chaining is like maximal chaining, except that when two non-strict inequalities are chained we split the resulting non-strict inequality into a strict inequality and an equality.

The calculus CS consists of the above inference rules plus irreflexivity resolution. (Maximal chaining is replaced by inequality chaining.)

Theorem 5. *Let N be a set of clauses that is saturated up to PO-redundancy (with respect to CS). Then N has a total ordering with equality as a model if and only if it does not contain the empty clause.*

[7] Multiple copies of an equation $u_i\sigma \approx t_1\sigma$ resulting from different inequalities $t_j < v_j$ and $t_k < v_k$ can of course be merged into one.

The variable elimination rule applies as previously, but the presence of equality complicates matters somewhat. To sum up what is explained in detail in our full paper, variable elimination is not necessarily a simplification rule, as some instances of a premise may not be rendered redundant by the conclusion. Fortunately, those instances of a clause, in which an unshielded variable x is instantiated by a maximal term, do become redundant as a result of variable elimination. Therefore, inferences involving the unshielded variable (chaining through x, superposition from x, [ir]reflexivity resolution applied to [in]equalities with x) are unnecessary. (The ordering constraints for such inferences require x to be instantiated with the maximal term, in which case the corresponding instances of the premises are redundant.) Hence variable chaining is not needed.

5 Experimental Results

We have implemented a theorem prover based on the above inferences systems (with explicit equality) which can handle arbitrary transitive relations, not just orderings and equality, cf. Bachmair and Ganzinger (1993b). The implemented inference systems also allow one to define selection functions on negative literals, so that negative chaining inferences need to be computed only with maximal or selected literals. The completeness of selection can be proved in a similar way as for superposition (Bachmair and Ganzinger 1994).

Our implementation is based on the Saturate system by Nivela and Nieuwenhuis (1993) which implements superposition together with a variety of simplification and elimination techniques. The system attempts to detect those properties of a given clause set (transitivity, reflexivity, irreflexivity, symmetry, totality, density, no endpoints, monotonicity) that are central to the selection of the appropriate instance of our family of chaining-based inference systems. Apart from implementing the inference rules we have also extended some of the mechanisms for redundancy proofs to a specific handling of inequations. For instance, we have extended the equational tautology checker to inequations, applying ground versions of the chaining inference systems as decision procedures to that end. We also reduce newly generated formulas containing inequational atoms by unit equations as well as inequations. (Reduction by negative chaining with unit inequations is sound if a tautology can be obtained that way.)

The system, which is implemented in Prolog, performs quite well on many non-trivial examples, but is a prototype in many respects. It is not at all tuned to fast proof search and there are no sophisticated inference selection schemes, clause weightings or indexing data structures. Its major purpose is to serve as a test-bed for trying out various combinations of chaining-based inference systems with ordering constraints and checks for redundancy.

The table below shows some benchmarks: ivt refers to the intermediate value theorem; scf to the the proof of continuity of the sum of two continuous functions (challenge problems 3 and 5, respectively, of Bledsoe 1990); and perm to one of the main lemmas about the composition of permutations of subarrays that one needs for the verification of quicksort. The last problem is interesting as it combines equality (to represent permutations as bijective functions) with inequality (for reasoning about the index ranges of subarrays).

problem	length	inferences	non-red	time	simpl	constraints
ivt	65	636	152	76	46	11
scf3	47	343	166	93	44	10*
scf5	56	858	268	240	150	21*
perm	67	450	210	143	40	79
perm	70	805	252	96	55	6*

length: proof length (including axioms)
inferences: number of computed inferences
non-red: number of computed inferences that were not proved to be redundant
time: total runtime in seconds on a Sparc10 under SICStus
simpl: time spent on simplification and redundancy proofs
constraints: time spent on solving ordering constraints

A central component of the system is a constraint solver for the lexicographic path ordering. Constraint solving can at times be very slow (e.g., in perm), as the underlying problem is NP-complete (Nieuwenhuis 1993). In fact this part can be prohibitively expensive, so that our system provides a choice between a full and a partial constraint solving mode. In the latter mode the ordering constraints on premises of inferences are checked individually, *before* a unifier is applied ("a priori constraints"). A (*) indicates runs with a priori constraint checking. These usually generate substantially more inferences, but the simpler constraint solving process may still result in a better overall runtime, as can be seen from the two runs of the perm example.

Ordering restrictions considerably cut down the search space of chaining, but ordered chaining by itself would still be too prolific and not enough goal-oriented to get any of these examples in reasonable time. The power of the prover stems from its combination of order-constrained inferences together with simplification and elimination of redundant clauses and inferences. From the benchmarks one observes that among the inferences that satisfy the ordering constraints, only 25%–45% are non-redundant ("non-red") with respect to the applied criteria. (But simplification and checking for redundancy usually is much more expensive than computing inferences.) Our abstract notion of redundancy is also of great value in checking whether certain simplification techniques are compatible with the inference system at hand. For instance for proving ivt it turned out that a simplification called resolution subsumption was extremely useful. To show that it was compatible and to add it to the system was very easy.

Our prover performs especially well in cases where transitive relations do not interfere too much with the algebraic structure of function symbols. This is the case for most of the lemmas that are needed for verifying sorting programs. At present, our system does not perform well on examples, such as scf5, with associative-commutative operators, such as addition, that are monotonic with respect to the ordering. Associativity and commutativity are dealt with by ordered rewriting, which is not a very practical method, despite the fact the our system applies specific checks for ground confluence of derived equations in such a case. We got the proof for scf5 only by using AC-unification instead of syntactic unification. But with this modification our implementation is incomplete as the lexicographic path ordering is not AC-compatible and as we do not generate extended clauses. In fact, we even had to slightly modify the problem formulation and run the problem with a priori constraints to make up for this incompleteness. An open problem which we are currently working on is how to combine decision methods for theories such as linear arithmetic with general chaining inferences. Once that has been worked out, there should be no need to resort to ad hoc techniques and problem massaging in examples such as scf5.

6 Summary

We have presented chaining-based inference systems for total orderings that extend previous work in several respects: we impose ordering constraints on chaining inferences; we build equality directly into the inference mechanism; and we establish refutational completeness in the presence of a notion of redundancy that covers such important simplification techniques as tautology deletion, subsumption, condensement, and demodulation. The completeness proofs of our results are comparatively simple. We deal with variable elimination as a simplification rule, separately from the chaining rules; an approach that better clarifies the connection between these essential components of chaining systems.

The improvements are not only of theoretical significance, as experimental evidence indicates that equational inference mechanisms, such as superposition and demodulation, are preferable to explicit axiomatizations of equality. Some form of equational reasoning appears to be used in the provers described by Bledsoe and Hines (1980), Hines (1988) and Hines (1990), and inference systems with equational reasoning capabilities may provide a better approximation to actual implementation practice than other chaining systems.

The superposition calculus, as described above, contains an explicit factoring rule for equalities, whereas maximal chaining implicitly encodes factoring for inequalities. An alternative to maximal chaining would be ordered chaining (as in Bachmair and Ganzinger 1993b) with inequality factoring.

In algebraic structures such as ordered rings one may have monotonicity or anti-monotonicity of functions (such as $+$ and $-$) with respect to the ordering. The techniques we have discussed should be extended to such cases. An important question in this context is to what extent chaining through or below variables is necessary. Our techniques may also be useful in the context of chaining and variable elimination for set theory, an application that has been studied by Hines (1990).

Acknowledgements. We would like to thank the anonymous referees for their comments.

References

L. BACHMAIR AND H. GANZINGER, 1990. On Restrictions of Ordered Paramodulation with Simplification. In M. Stickel, editor, *Proc. 10th Int. Conf. on Automated Deduction, Kaiserslautern*, Lecture Notes in Computer Science, vol. 449, pp. 427–441, Berlin, Springer-Verlag.

L. Bachmair and H. Ganzinger, 1993. Ordered Chaining for Total Orderings. Technical Report MPI-I-93-250, Max-Planck-Institut für Informatik, Saarbrücken. Full version of this paper.

L. Bachmair and H. Ganzinger, 1993. Rewrite Techniques for Transitive Relations. Technical Report MPI-I-93-249, Max-Planck-Institut für Informatik, Saarbrücken. To appear in Proc. LICS'94.

L. Bachmair and H. Ganzinger, 1994. Rewrite-based equational theorem proving with selection and simplification. *Journal of Logic and Computation*, Vol. 4, No. 3, pp. 1–31. Revised version of Technical Report MPI-I-91-208. To appear.

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

W. W. Bledsoe and L. M. Hines, 1980. Variable Elimination and Chaining in a Resolution-based Prover for Inequalities. In Wolfgang Bibel, Robert Kowalski, editors, *5th Conference on Automated Deduction*, LNCS 87, pp. 70–87, Les Arcs, France, Springer-Verlag.

W.W. Bledsoe, 1990. Challenge problems in elementary calculus. *Journal of Automated Reasoning*, Vol. 6, pp. 341–359.

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

L. Hines, 1990. Str$\dot{+}$ve$\subseteq$: The Str$\dot{+}$ve-based Subset Prover. In Mark E. Stickel, editor, *10th International Conference on Automated Deduction*, LNAI 449, pp. 193–206, Kaiserslautern, FRG, Springer-Verlag.

L. M. Hines, 1992. Completeness of a Prover for Dense Linear Logics. *Journal of Automated Reasoning*, Vol. 8, pp. 45–75.

L.M. Hines, 1988. Hyper-chaining and knowledge-based theorem proving. In *Proc. 9th Int. Conf. on Automated Deduction*, Lecture Notes in Computer Science, pp. 469–486. Springer-Verlag.

J. Hsiang and M. Rusinowitch, 1991. Proving refutational completeness of theorem proving strategies: The transfinite semantic Tree method. *Journal of the ACM*, Vol. 38, No. 3, pp. 559–587.

J. Levy and J. Agustí, 1993. Bi-rewriting, a Term Rewriting Technique for Monotonic Order Relations. In C. Kirchner, editor, *Rewriting Techniques and Applications*, Lecture Notes in Computer Science, vol. 690, pp. 17–31, Berlin, Springer.

R. Nieuwenhuis, 1993. Simple LPO constraint solving methods. *Information Processing Letters*, Vol. 47, No. 2, pp. 65–69.

P. Nivela and R. Nieuwenhuis, 1993. Saturation of first-order (constrained) clauses with the *Saturate* system. In Claude Kirchner, editor, *Rewriting Techniques and Applications, 5th International Conference, RTA-93*, LNCS 690, pp. 436–440, Montreal, Canada, Springer-Verlag.

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

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

J. R. Slagle, 1972. Automatic theorem proving for theories with Built-in Theories Including Equality, Partial Orderings, and Sets. *Journal of the ACM*, Vol. 19, pp. 120–135.