Basic Paramodulation

Leo Bachmair* Harald Ganzinger†
Christopher Lynch‡ Wayne Snyder§

March 9, 1995

*Department of Computer Science, SUNY at Stony Brook, Stony Brook, NY 11794, U.S.A., leo@sbcs.sunysb.edu; partially supported by NSF Grant No. CCR-8901322.

†Max-Planck-Institut für Informatik, Im Stadtwald, D-66123 Saarbrücken, Germany, hg@mpi-sb.mpg.de; partially supported by the ESPRIT Basic Research Working Group No. 6028 "Construction of Computational Logics" and by German Science Foundation Grant No. Ga261/4-2.

‡Computer Science Department, Boston University, 111 Cummington St., Boston, MA 02215, U.S.A., lynch@cs.bu.edu; partially supported by NSF grant No. CCR-8901647.

§Computer Science Department, Boston University, 111 Cummington St., Boston, MA 02215, U.S.A., snyder@cs.bu.edu; partially supported by NSF Grant No. CCR-8910268.

Proposed Running Head: **Basic Paramodulation**

Proofs should be sent to Wayne Snyder (Boston University—address on previous page).

List of Symbols

Other than ordinary text in variously sized fonts, and in roman, italic, boldface, and script ($\mathcal{F}$, $\mathcal{V}$, $\mathcal{R}$), greek letters, ordinary mathematical symbols ($\cup$, $\in$, $\notin$, $/$, $\setminus$ (set difference), $\neg$, $\vee$, $\wedge$, etc.) and the "at-sign" `@` used in email addresses, we use the following mathematical symbols:

- Relational symbols: $\approx$ (equality in the formal language), $=$ (equality in the meta language), $\not\approx$ (negation of equality), $\sim$ (congruence), $\Rightarrow$, $\Rightarrow^*$, $\Leftrightarrow$, $\Leftrightarrow^*$, $\Downarrow$ (rewrite relations), $\Rightarrow_{R\succ}$, $\Rightarrow_R$, etc. (subscripted rewrite relations);
- Term, Equation, and Clause Orderings: $\succ$, $\succeq$, $\prec$, $\succ_{mul}$, etc. (we use $>$ for the ordinary ordering on the natural numbers);
- Grouping symbols: $[\cdot]$, $[\cdot]_p$, $[\![\cdot]\!]$, $\{\cdot\}$,
- Miscellaneous: $\top$ (for **true**), $\mapsto$, and $\cdot$ (dot) used in closures: $C \cdot \sigma$.

We also use standard form for inferences, e.g.,

$$\frac{C_1 \cdot \rho \ \ \cdots \ \ C_n \cdot \rho}{C \cdot \theta}$$

and we indicate certain specific portions of an atomic formula by enclosing them in boxes, e.g.,

$$\neg P(gg\boxed{y}) \vee Q(ga) \vee k(hg\boxed{y}, g\boxed{y}) \approx h\boxed{y}$$

Abstract

We introduce a class of restrictions for the ordered paramodulation and superposition calculi (inspired by the *basic* strategy for narrowing), in which paramodulation inferences are forbidden at terms introduced by substitutions from previous inference steps. In addition we introduce restrictions based on term selection rules and redex orderings, which are general criteria for delimiting the terms which are available for inferences. These refinements are compatible with standard ordering restrictions and are complete without paramodulation into variables or using functional reflexivity axioms. We prove refutational completeness in the context of deletion rules, such as simplification by rewriting (demodulation) and subsumption, and of techniques for eliminating redundant inferences.

1 Introduction

The paramodulation calculus is a refutational theorem proving method for first-order logic with equality, originally presented in Robinson & Wos (1969) and refined in various ways since that time. Two important refinements of this method that have been developed are, first, restricting the paramodulation rule so that no inferences are performed into variable positions and avoiding the use of functional reflexivity axioms (Brand 1975, Peterson 1983) and, second, restricting the inference rules using orderings on terms and atoms (see Section 3.1 for references). In addition, various mechanisms have been suggested for simplifying clauses and removing redundant ones. The paramodulation rule is extremely prolific, even if restricted to non-variable positions, and it is crucial for the practical use of the method to work out the various possibilities for reducing the search space for a refutation.

In this paper we strengthen previous refinements significantly by extending the principles underlying the *basic* strategy for narrowing, due to Hullot (1980), in which inferences are forbidden at terms introduced by substitutions in earlier inferences, to the case of first-order clauses in a refutational setting. In addition, we show how to associate with each term information as to which subterms have already been explored, so as to direct further inferences to the unexplored region of a term. The boundary between the two regions is called the *frontier*. Theorem proving can be viewed as a process that continually expands this frontier in the search for a refutation. Our refinements of paramodulation are aimed at controlling and optimizing this exploration process.

As a simple illustration, let us consider the paramodulation inference

$$\frac{Q(ga) \lor f(hz, z) \approx gz \qquad \neg P(f(x, gy)) \lor k(x, gy) \approx hy}{\neg P(ggy) \lor Q(ga) \lor k(hgy, gy) \approx hy}$$

and possible further paramodulations into its conclusion. Using boxes to indicate subterms that have already been explored and at which further paramodulations are forbidden, we obtain the following representation of the conclusion

$$\neg P(gg\boxed{y}) \lor Q(ga) \lor k(hg\boxed{y}, g\boxed{y}) \approx h\boxed{y}$$

if paramodulations into variables are disallowed. The *basic* restriction also forbids inferences at any term introduced as part of the substitution,

$$\neg P(g\boxed{gy}) \lor Q(ga) \lor k(\boxed{hgy}, g\boxed{y}) \approx h\boxed{y}.$$

These restrictions can be implemented easily either by using a simple marking strategy (with a Boolean flag indicating forbidden terms) or, alternately, by directly implementing the formalism of closures (i.e., pairs of clauses and substitutions) in which we describe our inference systems. Alternately, as described in Nieuwenhuis & Rubio (1992a and 1992b), the basic strategy can be represented by clauses with equality constraints, e.g., $P(\boxed{a})$ could be represented by $P(x)\,[\![x = a]\!]$, where inferences are not permitted into the constraint (see Kirchner, Kirchner, & Rusinowitch 1990). This formulation also allows the integration of the basic strategy with other constraint methods.

We also show that the basic strategy is compatible with ordering restrictions and, hence, can be applied to the *superposition* calculus (see Bachmair & Ganzinger 1994) which extends a suitable notion of rewriting to first-order clauses. Further refinements include the use of *term selection functions* and *redex orderings*. Selection complements basic constraints in that it provides a mechanism for specifying at which positions inferences must take place, and is a generalization of the use of orderings to constrain inferences. Redex orderings blend well with selection functions and rest on the observation that the rewrite steps modelled by superposition can be assumed to have occurred in a particular order in reducing selected terms to normal form.

These refinements would allow us, for example, to forbid inferences at any term positioned below a former paramodulation inference,

$$\neg P(\boxed{ggy}) \vee Q(ga) \vee k(\boxed{hgy}, g\boxed{y}) \approx h\boxed{y}$$

or even at any term introduced by the left premise,

$$\neg P(\boxed{ggy}) \vee Q(\boxed{ga}) \vee k(\boxed{hgy}, g\boxed{y}) \approx h\boxed{y}.$$

We will also formally describe a technique, called *variable abstraction*, for propagating information about forbidden terms around a clause. For example, if one occurrence of a subterm has been explored, we may propagate the restrictions to other occurrences of the same term,

$$\neg P(\boxed{ggy}) \vee Q(\boxed{ga}) \vee k(\boxed{hgy}, \boxed{gy}) \approx h\boxed{y}.$$

The combined effect of all these refinements of paramodulation is reminiscent of the *set of support* strategy in resolution, in that inferences are not permitted in certain regions of the clause set. The differences lie in the scope of these restricted regions: parts of terms local to one clause in the basic method, and a subset of the entire clause set in the set of support strategy. Thus we consider this paper to be a robust answer to a research problem posed in Wos (1988): *What strategy can be used to restrict paramodulation at the term level to the same degree that the set of support strategy restricts all inference rules at the clause level*?

Another aspect of paramodulation calculi, which is at least as important for practical purposes as refinements of the deduction process, is the design of suitable simplification techniques. We explore the role of simplification rules such as demodulation, subsumption and blocking, and adapt the framework of redundancy developed in Bachmair & Ganzinger (1994) to our basic variants of paramodulation. The connections between simplification and deductive inference rules are quite subtle in this context and raise a number of interesting questions, both from a theoretical and a practical point of view.

This paper is organized as follows. In the next section we present the technical background to the calculi, which are presented formally in Section 3. The succeeding section proves completeness, and then we consider theorem proving derivations for saturating a set of clauses and discuss redundancy in Section 5. In Section 6 we will briefly consider the purely equational case and apply our results to describe Knuth/Bendix completion under the basic strategy. We conclude with a comparison with previous and current work.

2 Preliminaries

2.1 Equational clauses

We formulate our inference rules in an equational framework and define clauses in terms of multisets. A *multiset* is an unordered collection with possible duplicate elements. We denote the number of occurrences of an object x in a multiset M by $M(x)$.

An *equation* is an expression $s \approx t$, where s and t are (first-order) terms built from a given set of function symbols $\mathcal{F}$ and a set of variables $\mathcal{V}$. We assume the reader is familiar with some notation, such as strings of integers, for indicating *positions* (i.e., addresses of subterms) in a term, literal, or clause. By t/q we denote the subterm of t occurring at position q. We also write $t[s]$ if s is a subterm of t, and if necessary write $t[s]_p$ to indicate the position p of s in t. We identify $s \approx t$ with $t \approx s$ (and hence implicitly have symmetry of equality). A *literal* is either an equation A (a *positive* literal) or the negation $\neg A$ thereof (a *negative* literal). Negative equations $\neg(s \approx t)$ will be given in the form $s \not\approx t$. We may, where appropriate, assume the vocabulary of function symbols to be a many-sorted signature with the usual typing constraints for equations, terms and substitutions. In particular, atomic formulas $P(t_1, \ldots, t_n)$, where P is a "predicate" symbol, can be represented as equations $P(t_1, \ldots, t_n) \approx \top$, where $\top$ is a unary symbol of sort "atom" and the signature of predicate symbols is defined accordingly. For simplicity, we usually abbreviate $P(t_1, \ldots, t_n) \approx \top$ by $P(t_1, \ldots, t_n)$.

By a *ground* expression (a term, equation, literal, formula, etc.) we mean an expression containing no variables. A *clause* is a (finite) multiset of literals $\{L_1, \ldots, L_n\}$, which we usually write as a disjunction $L_1 \vee \ldots \vee L_n$.[1] A clause which is true in any (equality) interpretation is called a *tautology*. Examples of tautologies are clauses containing complementary literals (that is, literals A and $\neg A$) or containing an equation $t \approx t$.

A *substitution* is a mapping from variables to terms which is almost everywhere equal to the identity. By $E\sigma$ we denote the result of applying the substitution σ to an expression E and call $E\sigma$ an *instance* of E. If $E\sigma$ is ground, we speak of a *ground instance*. For example, the clause $a \approx b \vee a \approx b$ is an instance of $x \approx b \vee a \approx y$. Composition of substitutions is denoted by juxtaposition. Thus, if τ and ρ are substitutions, then $x\tau\rho = (x\tau)\rho$, for all variables x. We define $dom(\sigma) = \{x | x\sigma \neq x\}$. If θ and σ are two substitutions such that $dom(\theta) \cap dom(\sigma) = \emptyset$, then we define their *union*, denoted $\theta + \sigma$, as the substitution which maps x to $x\theta$ if $x\theta \neq x$, and to $x\sigma$ otherwise.

2.2 Equality Herbrand interpretations

Because we formulate our system wholly in an equational framework, we may represent Herbrand interpretations as congruences on ground terms. We write $A[s]$ to indicate that A contains s as a subexpression and (ambiguously) denote by $A[t]$ the result of replacing a particular occurrence of s by t. An *equivalence* is a reflexive, transitive, symmetric binary relation. An equivalence $\sim$ on terms is called a *congruence* if $s \sim t$ implies $u[s] \sim u[t]$, for all terms u, s, and t. If E is a set of ground equations, we denote by E^* the smallest congruence $\sim$ such that $s \sim t$ whenever $s \approx t \in E$.

[1] Therefore we assume that the order of the literals in a disjunction is unimportant, i.e., $A \vee B$ is the same clause as $B \vee A$; also note that $A \vee A$ is distinct from A.

By an (*equality Herbrand*) *interpretation* we mean a congruence on ground terms. An interpretation I is said to *satisfy* a ground clause C if either $A \in I$, for some equation A in C, or else $A \notin I$, for some negative literal $\neg A$ in C. We also say that a ground clause C is *true in* I, if I satisfies C, and that C is *false in* I otherwise. An interpretation I is said to satisfy a non-ground clause C if it satisfies all ground instances $C\sigma$. For instance, a tautology is satisfied by any interpretation. The empty clause is *unsatisfiable* in that it is satisfied by no interpretation. An interpretation I is called a (*equality Herbrand*) *model* of a set N of clauses if it satisfies all members of N. A set N is called *consistent* if it has a model; and *inconsistent* (or *unsatisfiable*), otherwise. We say that a clause C is a *consequence of* N if every model of N satisfies C.

Convergent rewrite systems provide a convenient formalism for describing and reasoning about equality interpretations.

2.3 Convergent rewrite systems

A binary relation $\Rightarrow$ on terms is called a *rewrite relation* if $s \Rightarrow t$ implies $u[s\sigma] \Rightarrow u[t\sigma]$, for all terms s, t and u, and substitutions σ. It is called *well-founded* if there is no infinite sequence $t_0 \Rightarrow t_1 \Rightarrow \cdots$. A transitive, well-founded rewrite relation is called a *reduction ordering.* By $\Leftrightarrow$ we denote the symmetric closure of $\Rightarrow$; by $\Rightarrow^*$ the transitive, reflexive closure; and by $\Leftrightarrow^*$ the symmetric, transitive, reflexive closure. Furthermore, we write $s \Downarrow t$ to indicate that s and t can be rewritten to a common form: $s \Rightarrow^* v$ and $t \Rightarrow^* v$, for some term v. A rewrite relation $\Rightarrow$ is said to be *Church-Rosser* if the two relations $\Leftrightarrow^*$ and $\Downarrow$ are the same.

A set of equations R is called a *rewrite system* with respect to an ordering $\succ$ if we have $s \succ t$ or $t \succ s$, for all equations $s \approx t$ in R. If all equations in R are ground, we speak of a ground rewrite system. Equations in R are also called (*rewrite*) *rules.* When we speak of "the rule $s \approx t$" we implicitly assume that $s \succ t$. By $\Rightarrow_{R\succ}$ (or simply $\Rightarrow_R$) we denote the smallest rewrite relation for which $s \Rightarrow_R t$ whenever $s \approx t \in R$ and $s \succ t$. A term s is said to be in *normal form* (with respect to R) if it can not be rewritten by $\Rightarrow_R$, i.e., if there is no term t such that $s \Rightarrow_R t$. A term is also called *irreducible*, if it is in normal form, and *reducible*, otherwise. For instance, if $s \Downarrow_R t$ and $s \succ t$, then s is reducible by R. A substitution σ is called *normalized* with respect to R if $x\sigma$ is in normal form for each $x \in dom(\sigma)$.

A rewrite system R is said to be *convergent* if the rewrite relation $\Rightarrow_E$ is well-founded and Church-Rosser. Convergent rewrite systems define unique normal forms. A ground rewrite system R is called *left-reduced* if for every rule $s \approx t$ in R the term s is irreducible by $R \setminus \{s \approx t\}$. It is well-known that left-reduced, well-founded ground rewrite systems are convergent (see Huet 1980).

We shall represent equality Herbrand interpretations in this paper by convergent ground rewriting systems. Any such system R represents an interpretation I defined by: $s \approx t$ is true in I iff $s \Downarrow_R t$. Thus we shall use the phrase "is true in R" instead of the more proper "is true in the interpretation I generated by R."

2.4 Clause orderings

In this paper we assume given a reduction ordering $\succ$ which is total on ground terms.[2] For the purpose of extending this ordering to literals and clauses, we identify a positive literal $s \approx t$ with the multiset (of multisets) $\{\{s\}, \{t\}\}$, and a negative literal $s \not\approx t$ with the multiset $\{\{s, t\}\}$.

Any ordering $\succ$ on a set S can be extended to an ordering $\succ_{mul}$ on finite multisets over S as follows: $M \succ_{mul} N$ if (i) $M \neq N$ and (ii) whenever $N(x) > M(x)$ then $M(y) > N(y)$, for some y such that $y \succ x$. If $\succ$ is a total [well-founded] ordering,[3] so is $\succ_{mul}$. Given a set (or multiset) S and an ordering $\succ$ on S, we say that x is *maximal* relative to S if there is no $y \in S$ with $y \succ x$; and *strictly maximal* if there is no $y \in S \setminus \{x\}$ with $y \succeq x$.

If $\succ$ is an ordering on terms, then the twofold multiset ordering $(\succ_{mul})_{mul}$ of $\succ$ is an ordering on literals, and the threefold ordering $((\succ_{mul})_{mul})_{mul}$ is an ordering on clauses. Note that the multiset extension of a well-founded [total] ordering is still well-founded [total]. Since which ordering we intend will always be clear from the context, we denote all of these simply by $\succ$. When comparing a literal with a clause, we consider the literal to be a unitary clause. These orderings are similar to the ones used in Bachmair & Ganzinger (1994). For example, if $s \succ t \succ u$, then $s \not\approx u \succ s \approx t \succ s \approx u$. In general, $\neg A \succ A$, for all equations A.

In the setting in which we work we need a notion of reducibility which takes account of the ordering on the literals involved. We say that a literal $L[s']_p$ is *order-reducible* (at position p) by an equation $s \approx t$, if $s' = s\rho$, $s\rho \succ t\rho$ and $L \succ s\rho \approx t\rho$. The last condition is always true when L is a negative literal or else when the redex s' does not occur at the top of the largest term of L. For example, if $c \succ b \succ a$, then $c \approx b$ is order-reducible by

[2] We assume the implicit unary predicate $\top$ is least in this ordering.

[3] We shall often abbreviate the parenthetical "(respectively, ...)" by "[...]".

$c \approx a$, and $c \not\approx a$ is order-reducible by $c \approx b$, but $c \approx a$ is not order-reducible by $c \approx b$. Note that no equation is order-reducible by itself. But a ground instance of an equation may be order-reducible by another ground instance of the same equation, as the above two ground instances of $c \approx x$ indicate. A literal is order-reducible by R if it is order-reducible by some equation in R. Likewise, a clause is called order-[ir]reducible at p if the literal to which p belongs is order-[ir]reducible at p. "Order-irreducible" is the same as "not order-reducible."

2.5 Closures

Basic strategies require additional information about the terms in a clause. A *frontier* for a term t is a set of mutually disjoint positions in t. We assume that frontiers are associated with all terms in a clause. Paramodulation inferences will be forbidden at any term at or below a frontier position. Thus, each term is effectively divided into an *explored region* (all positions at or below some frontier position) and an *unexplored region* (all remaining positions). When displaying formulas we use boxes, as in the examples above, to delineate the explored regions in terms. Our proposed restrictions on paramodulation inferences are designed to maximize the explored regions, as this cuts down the number of inferences that can be applied to a clause. The fundamental observation underlying the basic strategy is that frontier positions need not be retried when clauses are instantiated via unifiers during the deductive inference process.

A *closure* is a pair $C \cdot \sigma$ consisting of a clause C (the *skeleton*) and a substitution σ. Closures provide a convenient formalism for denoting clauses and associated frontiers: $C \cdot \sigma$ represents the clause $C\sigma$ with frontiers consisting of all positions of variables x in C for which $x\sigma \neq x$. For example,

$$(P(x) \vee z \approx b) \cdot \{x \mapsto fy, z \mapsto gb\}$$

is a closure representing the clause $P(fy) \vee gb \approx b$, but which we will conventially represent as $P(\boxed{fy}) \vee \boxed{gb} \approx b$. A non-variable position p in $C\sigma$ is called a *substitution position* in $C \cdot \sigma$ if it can be written as $p = p'q$, where p' is a variable position in C. In our previous example, the term fy occurs at a substitution position, but y does not. The term b occurs twice, once at a substitution position.

We will occasionally extend this notation to terms, equations, and subsets of clauses, e.g., representing a term occurring in a closure $C \cdot \sigma$ by $t \cdot \sigma$.

We speak of a *ground closure* if $C\sigma$ is ground. The closure $C \cdot \mathrm{id}$, where id is the identity substitution, represents the clause C with no associated frontier. An *instance* $C \cdot \sigma\rho$ of a closure $C \cdot \sigma$ (by a substitution ρ) represents the clause $C\sigma\rho$. A closure $C\sigma_1 \cdot \sigma_2$ is called a *retraction* of $C \cdot \sigma$ if $\sigma = \sigma_1\sigma_2$.[4] When a retraction is formed, we assume that any variables introduced are new. For example

$$(P(x) \vee gz' \approx b) \cdot \{x \mapsto fy, z' \mapsto b\}$$

is a retraction of the closure given in the previous paragraph.

We say that two closures $C \cdot \sigma$ and $D \cdot \tau$ *have disjoint variables* whenever $var(C) \cup var(C\sigma)$ and $var(D) \cup var(D\tau)$ are disjoint. In this case $C \cdot \sigma$ and $D \cdot \tau$ represent the same clauses and frontiers as $C \cdot \rho$ and $D \cdot \rho$, respectively, where $\rho = \sigma\tau$.

Since we are only interested in the clauses and frontiers represented by a closures, the latter may be kept in a certain form during a refutation. Let us say that a closure $C \cdot \sigma$ is in *standard form* if for every variable x occurring in C, either $x\sigma = x$ or $x\sigma$ is a non-variable. For example, the closures given above are in standard form, whereas

$$P(fx, z) \cdot \{x \mapsto y, z \mapsto y\}$$

is not. We will assume in what follows that all closures are kept in standard form by instantiating variable–variable bindings whenever they arise. This is merely a technical convenience and has no effect on the restrictions discussed in the paper.

2.6 Reduced Closures

The main technical problem in completeness proofs for paramodulation systems is that ground inferences on ground instances of clauses (which is the level where the fundamental properties related to completeness are proved) do not necessarily "lift" to corresponding inferences on the clauses themselves, as the position of the inference may be lifted off with the substitution. The solution to this, due to Peterson (1983), has been to work with substitutions which are reduced with respect to a suitably defined rewrite system constructed from the set of ground instances of clauses; in our method we carry this one step further and require that clauses be "hereditarily reduced," so that no inference need be performed inside *any* substitution position. In

[4] In [23, 24] this is called a *weakening*.

other words, the restriction that inferences not be performed at variable positions in premises is inherited by the conclusions of inferences, so that no inference need be performed at or below a position where a variable has ever occurred during the accumulation of substitution terms. The key to formalizing this approach is a suitable notion of what it means for a closure to be reduced.[5]

We say that a ground closure $C \cdot \sigma$ is *reduced* with respect to a rewrite system R (or *R-reduced*) *at a position p* if $C\sigma$ is order-irreducible by R at or below p. The closure $C \cdot \sigma$ is simply called *reduced* with respect to R if it is reduced at all substitution positions.

For example $P(\boxed{fb}) \vee \boxed{fa} \approx a$ is reduced with respect to the system $\{fa \approx a\}$, but $\boxed{fa} \not\approx a$ is not.

A non-ground closure $C \cdot \sigma$ is called reduced with respect to R if for any of its ground instances $C \cdot \sigma\rho$ it is the case that $C \cdot \sigma\rho$ is reduced with respect to R whenever $C\sigma \cdot \rho$ is (e.g., when ρ is normalized with respect to R, then $C \cdot \sigma\rho$ will be reduced with respect to R). These definitions are extended to closure literals in the obvious way. Note that closures $C \cdot \mathrm{id}$ with an empty substitution part are reduced with respect to any rewrite system R. A ground clause D is called a reduced ground instance (with respect to R) of a set N of closures if there exists a closure $C \cdot \sigma$ in N such that $D = C\sigma\tau$ and $C \cdot \sigma\tau$ is reduced with respect to R.

3 Basic Inference Rules

We shall consider inference rules of the form

$$\frac{C_1 \cdot \rho \quad \cdots \quad C_n \cdot \rho}{C \cdot \theta}$$

where $n \in \{1, 2\}$ and $C_1 \cdot \rho, \ldots, C_n \cdot \rho$ (the *premises*) and $C \cdot \theta$ (the *conclusion*) are closures. We assume that the premises of a binary inference rule have disjoint variables (if necessary the variables in one of the premises are renamed with new variables), and so may give a common name ρ to their substitutions for notational convenience.

[5] We should remark at this point that the main technical difficulties in formalizing the "basic" concepts arise only in the context of non-Horn clauses and in the presence of variables that occur in positive equations, but not as arguments of function symbols, in a clause. The exposition can be considerably simplified if one considers, say, only Horn clauses.

3.1 Basic paramodulation

The inference systems we discuss consist of restricted versions of paramodulation, equality resolution, and factoring. Let us first discuss *paramodulation* (Robinson & Wos 1966), the *basic* variant of which is:

$$\frac{(C \vee s \approx t) \cdot \rho \quad (L[u] \vee D) \cdot \rho}{(L[t] \vee C \vee D) \cdot \theta}$$

where the *redex* u is not a variable and $\theta = \rho\sigma$, where σ is a most general unifier[6] of $s\rho$ and $u\rho$. These are basic refinements of paramodulation in the sense that unifiers are composed with the substitution part of a closure but not applied to its skeleton and inferences do *not* take place at substitution positions (by virtue of the restriction "u is not a variable").

Since we formulate our rules in an equational framework, basic resolution inferences are a special case of basic paramodulation. For simplicity in the sequel we discuss only paramodulation, leaving the translation to the resolution case to the reader; see also Bachmair & Ganzinger (1994).

We next refine basic paramodulation along two parameters, first using a given reduction ordering $\succ$ to restrict the first premise, and second by the use of a term selection function which delimits the locations in the second premise where redexes can occur. Later on, we will in addition use a redex ordering to specify which selected positions in both premises can be assumed to be reduced.

The use of orderings may be motivated as follows. Assume given a reduction ordering $\succ$. We say that a clause $C \vee s \approx t$ is *reductive* for $s \approx t$ if $t \not\succeq s$ and $s \approx t$ is a strictly maximal literal in the clause. For example, if $s \succ t \succ u$, then $s \approx u \vee s \approx t$ is reductive for $s \approx t$, but $s \not\approx u \vee s \approx t$ is not. In general, if a clause C is reductive for $s \approx t$, then the maximal term s must not occur in a negative literal. If the reduction ordering $\succ$ is total on ground terms, then a reductive ground clause

$$\neg A_1 \vee \cdots \vee \neg A_m \vee B_1 \vee \cdots \vee B_n \vee s \approx t$$

can be thought of as a *conditional rewrite rule*

$$A_1, \ldots, A_m, \neg B_1, \ldots, \neg B_n \rightarrow s \approx t$$

[6] We assume in this paper that all most general unifiers are such as produced by the Martelli–Montanari set of transformations [32]; the reader may check that when the variables in the premises are disjoint, then all substitutions will be idempotent.

(with positive and negative conditions), where all conditions are strictly smaller than $s \approx t$.[7] Conditional rules of this form define a rewrite relation on ground terms ("replace s by t whenever all conditions are satisfied"), so that corresponding paramodulations on the ground level can be thought of as rewriting applied to ground clauses. Our completeness proof shows that constructing a refutation proof can (at the ground level) be seen as the process of partially constructing a convergent rewrite system from reductive clauses and normalizing negative equations to identities (which are thereupon removed).

Selection rules (generalized from Bachmair & Ganzinger 1994) define a minimal set of positions where inferences must be performed to achieve this end. We define a *term selection function* (or just a *selection function*) to be a function S that assigns to each closure C a set $S(C)$ of *selected* occurrences of non-variable terms in C, subject to the following constraints. Let us say that an occurrence of a literal in C is *selected* if it contains a selected occurrence of a term; then we require that (i) *some* negative equation or *all* maximal literals must be selected, and (ii) the maximal side(s) of a selected literal, and all its non-variable subterms, must be selected. Thus, if a negative equation in C is maximal, it must be selected.

Inferences may only take place at selected terms, but we should emphasize that a given selection rule may select more terms than are strictly required; below we shall see that there is an interesting tradeoff between the strength of the selection rule and the basic restriction. Finally, it should be remarked that with respect to negative equations, this strategy is much stronger than the usual ordering restrictions. In the latter, we must allow for redexes in all maximal equations, but according to our selection strategy, we need only select a *single* negative equation. This shows clearly the difference between the *don't care non-deterministic* choices which must be made in searching for a redex among the negatives namely, which negative equation to work on next, and the choices which are *don't know non-deterministic*, namely, which redex to pick in the selected term(s) in the chosen negative equation. Essentially, our results show that orderings are significant with regard to positive equations, since they guide the construction of critical pairs, but with negatives, orderings play a minimal role compared with selection functions, since (as in SLD-resolution) the choice of a negative atom to work on is don't care non-deterministic.

Based on these two methods for obtaining restrictions we get:

[7] These systems have been introduced and investigated by Kaplan (1988).

Basic paramodulation: $\dfrac{(C \vee s \approx t) \cdot \rho \quad (L[u] \vee D) \cdot \rho}{(L[t] \vee C \vee D) \cdot \theta}$

where (i) u is not a variable and $\theta = \rho\sigma$, where σ is a most general unifier of $s\rho$ and $u\rho$, (ii) the clause $C\theta \vee s\theta \approx t\theta$ is reductive for $s\theta \approx t\theta$ and contains no negative selected equations (thus $s\theta$ will be selected), (iii) $u\theta$ is a selected term in $L\theta \vee D\theta$, (iv) $L\theta \not\preceq C\theta \vee s\theta \approx t\theta$, and (v) if $t\theta$ is selected and L is a negative literal $u \not\approx v$, then $u\theta \approx v\theta \not\prec s\theta \approx t\theta$.

We emphasize that we use selection not only to control where inferences may take place, but also to disallow inferences where the first premise contains negative selected equations. It is this feature that allows us to achieve the effect of hyper-resolution and hyper-paramodulation strategies, cf. Bachmair & Ganzinger (1994).

For a paramodulation inference with premises $C_1 \cdot \rho$ and $C_2 \cdot \rho$ and conclusion $D \cdot \theta$ one typically can require that $C_1\theta \not\succeq C_2\theta$ and $D\theta \not\succeq C_2\theta$. The fourth condition we give above not only strengthens this restriction, but seems also easier to check in practice. These restrictions arise from the induction ordering used at the ground level in the completeness proof and require a more refined ordering on clauses, as in Zhang (1988), Bachmair & Ganzinger (1990) and Pais & Peterson (1991), rather than just an ordering on atoms, as in Peterson (1983) and Hsiang & Rusinowitch (1992).

The technique of selection rules for paramodulation can be used to simulate restrictions on redexes based on reduction orderings, such as standard paramodulation and superposition. For example, ordered paramodulation as it appears in Peterson (1983) or Hsiang & Rusinowich (1992) can be obtained via a selection rule which selects both sides of each maximal equation in a clause, and the superposition calculus of Bachmair & Ganzinger (1990) can be obtained by selecting all maximal sides of maximal equations (and using the *equality factoring* rule to be presented below). Positive paramodulation (i.e., the left premise can contain no negatives) is obtained if the rule always selects a negative equation if such exists. Also, certain results which have previously required special proofs are obtained as immediate corollaries of our main completeness theorem. For example, resolution is complete if no clause is ever resolved with itself (Eisinger 1989); in the paramodulation case, we can show that completeness is preserved if we forbid paramodulation of a clause into its own negative literals (but note that the construction of critical pairs must allow for the paramodulation of a clause into its own positive literals). This can easily be seen by considering a selection rule which is invariant under substitution (e.g., which is determined by the skeleton of a clause only) and never selects a positive and a negative equation simultaneously. In a later section we shall add further restrictions to paramodulation in the form of blocking rules.

In addition to paramodulation we need an inference rule that encodes the reflexivity of equality:

$$\textbf{Equality resolution:} \qquad \frac{(C \vee u \not\approx v) \cdot \rho}{C \cdot \theta}$$

where $\theta = \rho\sigma$, with σ a most general unifier of $u\rho$ and $v\rho$ and $u\theta \not\approx v\theta$ a selected literal in $C\theta \vee u\theta \not\approx v\theta$.

We also need a variant of factoring, restricted to positive literals:

$$\textbf{Equality factoring:} \qquad \frac{(C \vee s \approx t \vee s' \approx t') \cdot \rho}{(C \vee t \not\approx t' \vee s' \approx t') \cdot \theta}$$

where (i) $\theta = \rho\sigma$, with σ a most general unifier of $s\rho$ and $s'\rho$, (ii) $t\theta \not\succeq s\theta$ and $t'\theta \not\succeq s'\theta$, (iii) $s\theta \approx t\theta$ is a selected equation and no negative literal is selected in $C\theta \vee s\theta \approx t\theta \vee s'\theta \approx t'\theta$, and (iv) if $t\theta$ is selected then $t\theta$ and $t'\theta$ are unifiable.

Equality factoring is evidently sound, as the implication $t\theta \approx t'\theta \supset s'\theta \approx t'\theta$ is a logical consequence of the disjunction $s'\theta \approx t\theta \vee s'\theta \approx t'\theta$. An alternative to equality factoring is to use positive factoring plus the *merging paramodulation* rule of Bachmair & Ganzinger (1990), but the technical development for the current system is simpler.

3.2 Variable abstraction

Basic paramodulation, equality resolution, and equality factoring are our core inference rules. We will also employ an auxiliary inference rule in our calculus which can be applied to the conclusions of inferences for expanding the frontier of a new closure by moving skeleton terms into the substitution:

$$\textbf{Variable abstraction:} \qquad \frac{C[t]_p \cdot \sigma}{C[x]_p \cdot \{x \mapsto t\}\sigma}$$

where p is a non-variable position in C and x is a new variable. We also speak of a variable abstraction *at position p*.

Obviously, we lose completeness if this inference rule is applied at arbitrary positions, for then no paramodulation inferences may be possible at all. The problem is to find out at which positions variable elimination can be safely applied. The fundamental idea here, as mentioned in the introduction, is that it is possible to propagate certain "basic" restrictions on redexes to other occurrences of the same term; for example, $P(a, \boxed{a})$ can be abstracted to $P(\boxed{a}, \boxed{a})$, since (at the ground level) if one occurrence of a is reduced, then so is the other. In addition, it is possible to apply this rule during the construction of the conclusions of inferences, based on information about what terms (at the ground level) can be assumed to be reduced. Before we formalize this idea, we motivate the notion of a "redex ordering."

We have remarked above that paramodulation, on the ground level, corresponds to conditional rewriting, while its repeated application achieves normalization of ground clauses. In this interpretation, paramodulation into negative equations amounts to tracing rewrite proofs for the two sides of the equation, and paramodulation into positive equations serves to construct critical pairs, and, hence, to allow the construction of convergent rewrite systems (our completeness proof will be founded on this idea). Term selection defines which positions must be considered as possible redexes in this process. One important property of convergent systems is that any fair strategy for finding redexes, i.e., one which does not ignore a possible redex forever, can be used to normalize terms. For example, searching for redexes in depth-first, left to right order is fair in this sense. In general, one could define a function from terms to an ordering on positions in the term, and the normalization process could always use the ordering to search for redexes. In our setting, in fact it is possible to order the set of all positions occurring in selected terms in a closure; when a redex is selected, then it may be assumed that all positions lower in the ordering are in normal form; we may formalize this as follows.

Let $\mathcal{R}$ be a function which for any multiset M of (closure) terms returns a partial order on the positions in the selected terms in M. Thus, for any closure C, $\mathcal{R}(S(C))$ is an ordering on the positions in C where redexes are allowed in our paramodulation rules. We will call such an ordering a *redex ordering*, and denote it by $\prec_{\mathcal{R}}$ when S and C are obvious from context. We shall see that the ordering $\prec_{\mathcal{R}}$ serves to direct the search for a redex among disjoint innermost redexes in a term. (Therefore, it is only necessary to consider orderings which contain the subterm ordering on the terms in M, i.e., if $t[t'] \in M$, then $t \not\prec_{\mathcal{R}} t'$.)

The essential idea is that when a paramodulation inference is performed into a position q, then all selected positions $p \prec_{\mathcal{R}} q$ can be assumed to be reduced, and hence amenable to being moved into the substitution part of the conclusion using variable abstraction. Thus, redex orderings can be combined with selection functions to guide the variable abstraction process as applied to the conclusions of paramodulation inferences.

Formally, we say that a position p in the conclusion $C[t]_p{\cdot}\sigma$ of an inference is *eligible* for variable abstraction if, for any arbitrary rewrite system R for which $C\sigma$ is order-reducible at p, either (i) some premise or the conclusion itself is order-reducible by R at a substitution position, or (ii) the first (or only) premise is order-reducible by R at a selected position, or (iii) the second premise, in the case of a paramodulation inference applied at a position q, is order-reducible by R at a selected position that is disjoint from and smaller (with respect to $\prec_{\mathcal{R}}$) than q.

Variable elimination may be applied to eligible positions in the conclusions of inferences. These additional inferences are *optional*, that is, variable abstraction need not be applied to all eligible positions. Our completeness results apply to all strategies for applying variable abstraction at eligible positions. In practice, most eligible terms can be identified by checking for the existence of terms in suitable selected or substitution positions that are identical to skeleton terms in the conclusion.

The technique of redex orderings is a generalization of a similar technique used in narrowing (see Krischer & Bockmair 1991). Briefly, the reason this technique does not disturb refutational completeness is that in our proof we use the fact that substitutions can be kept in normal form (with respect to a suitable rewrite system), and so normalized terms can always be moved into the substitution. In addition, we may restrict (at the ground level) the first premise of a paramodulation inference, and the single premise of the unary inference rules, to those clauses in which selected terms are normalized, and may assume that selected terms in the second premise are to be normalized using the given redex ordering $\prec_{\mathcal{R}}$, so that all terms less than the redex are in normal form. Details will be given in the next section.

To summarize, we have defined a class of *basic inference systems* comprising equality resolution, equality factoring, and paramodulation, plus subsequent variable abstraction, which depend on the following parameters: a reduction ordering $\succ$, a selection function S, and a redex ordering function $\mathcal{R}$. Such inference systems embed four kinds of restrictions: (i) basic constraints preventing paramodulations into those parts of a clause generated by previous substitutions; (ii) ordering constraints allowing only paramodulations that approximate conditional rewriting (on the ground level); (iii) selection functions excluding paramodulations into non-selected terms and from clauses with selected negative equations; and (iv) redex orderings for defining the order in which inferences can be assumed to have occurred. Basic constraints define the frontier between explored and unexplored regions of a clause, while ordering constraints and selection are mechanisms for controlling the application of inferences at unexplored positions; redex orderings define conditions under which the frontier can be expanded in newly constructed closures. (A further technique for restricting inferences based on reducibility criteria will be presented in a later section.)

The soundness of the inference system presented in this section is straightforward and left to the interested reader. In the next section we prove that these basic calculi are refutationally complete in the sense that a contradiction (the empty clause) can be derived from any inconsistent set of clauses.

4 Refutational Completeness

We prove completeness by showing that if a set of closures N which is saturated with respect to our inference rules does not contain the empty closure, then it is possible to construct a model, represented by a convergent rewrite system, for N. This means that the empty closure can be derived from any inconsistent set of closures.

4.1 Construction of Equality Interpretations

Let N be a set of closures in standard form and recall that $\succ$ is assumed to be a reduction ordering which is total on ground terms. We define interpretations R by means of convergent rewrite systems as follows.

First, we use induction on the clause ordering $\succ$ to define sets of equations E_C and R_C, for all ground instances C of closures of N.

Definition 1 Let C be such a ground instance and suppose that $E_{C'}$ and $R_{C'}$ have been defined for all ground instances C' of N for which $C \succ C'$. Then

$$R_C = \bigcup_{C \succ C'} E_{C'}.$$

Moreover

$$E_C = \{s \approx t\}$$

if $C = D \vee s \approx t$ is a reduced ground instance of N with respect to R_C such that (i) C is false in R_C, (ii) C is reductive for $s \approx t$, and (iii) s is irreducible by R_C. In this case, we say that C *produces* the equation (or rule) $s \approx t$.

In all other cases, $E_C = \emptyset$. Finally, we define $R = \bigcup_C E_C$ as the set of all equations produced by ground instances of clauses of N.

Clauses that produce equations are called *productive*. Note that a productive clause C is false in R_C, but true in $R_C \cup E_C$. The sets R_C and R are constructed in such a way that they are left-reduced rewrite systems with respect to $\succ$. Hence, they are convergent, and so, as we have remarked previously, represent interpretations of the set of clauses N, and can also be used in conjunction with a redex ordering to normalize selected terms in a closure.

We shall also use the following ancillary results in our completeness proof.

Lemma 1 *Let $C = B \vee s \approx t$ be a ground instance of N where $s \approx t$ is a maximal occurrence of an equation, and let D be another ground instance of N containing s. If $C \succ D$ and s is irreducible by R_C, then $R_C = R_D$.*

Proof. If C' is any ground instance of N with $C \succ C' \succeq D$, then $E_{C'} = \emptyset$, for otherwise s would be reducible by R_C. Therefore $R_C = R_D \cup \bigcup_{C \succ C' \succeq D} E_{C'} = R_D$. □

Lemma 2 *Let $C = B \vee u \not\approx v$ and D be ground instances of N with $D \succeq C$. Then $u \approx v$ is true in R_C if and only if it is true in R_D if and only if it is true in R.*

Proof. If $u \approx v$ is true in R_C, then $u \Downarrow_{R_C} v$. Since $R_C \subseteq R_D \subseteq R$, we then have $u \Downarrow_{R_D} v$ and $u \Downarrow_R v$, which indicates that $u \approx v$ is true in R_D and in R.

On the other hand, suppose $u \approx v$ is false in R_C. If u' and v' are the normal forms of u and v with respect to R_C, then $u' \neq v'$. Furthermore, if $s \approx t$ is a rule in $R \setminus R_C$, then $s \succ u \succeq u'$ and $s \succ v \succeq v'$. (Clauses which produce rules for terms not greater than u or v are smaller than C.) Therefore, u' and v' are in normal form with respect to R, which implies that $u \approx v$ is false in R_D and in R. □

Lemma 3 *Let $C = B \vee u \approx v$ and D be ground instances of N with $D \succeq C$. If $u \approx v$ is true in R_C, then it is also true in R_D and in R.*

Proof. Use the fact that $R_C \subseteq R_D \subseteq R$. □

The above lemmas indicate that the sequence of interpretations R_C, with C ranging over all ground instances of N, preserves the truth of ground clauses.

Corollary 1 *Let C and D be ground instances of N with $D \succeq C$. If C is true in R_C, then it is also true in R_D and R.*

Next, we show that the property of being a reduced closure is also preserved.

Lemma 4 *A ground closure C is a reduced ground instance of N with respect to R_C if and only if it is reduced with respect to R.*

Proof. If C is not reduced with respect to R, then there is some clause D which produces an equation $s \approx t$, and some literal L in C which is reducible at a substitution position by $s \approx t$ and such that $s \approx t \prec L$. Since $s \approx t$ is strictly maximal in D, clearly $D \prec C$, and C is not reduced with respect to R_C. For the converse use the fact that $R_C \subseteq R$. □

Finally, it will be useful in a number of places to construct reduced closures in the following way.

Lemma 5 *Suppose $C \cdot \rho\sigma$ is a ground instance of a closure $C \cdot \rho$ in N. Then there is a ground instance $C \cdot \rho\tau$ such that (i) $C\rho\sigma \succeq C\rho\tau$, (ii) $C\rho \cdot \tau$ is reduced with respect to R, and (iii) $C\rho\tau$ is true in R_D [R] if and only if $C\rho\sigma$ is true in R_D [R], for any clause $D \succeq C\rho\sigma$.*

Proof. Define τ to be the substitution for which $x\tau$ is the normal form of $x\sigma$ by $R_{C\rho\sigma}$. Then (i) and (iii) are evidently satisfied. For (ii), since $C\rho \cdot \tau$ is reduced with respect to $R_{C\rho\sigma}$, then clearly it is reduced with respect to $R_{C\rho\tau}$, so then by the previous lemma it is reduced with respect to R. □

4.2 Redundancy and Saturation

We shall prove that the interpretation R is a model of N, provided N is consistent and saturated, i.e., closed under sufficiently many applications of the appropriate basic inference rules. In addition we shall demonstrate that the search space can be further decreased by certain restrictions which are based on the concept of redundancy. Roughly, a closure is redundant if it is a consequence of smaller closures in N. Such closures are unnecessary in saturating a set of closures, since they will play no role in the model construction given above. In addition, it is possible to show that certain inferences are redundant as well, in that the conclusions of such inferences will play no role in the model construction.

For any ground clause C and set of clauses N, let N_C be the set of ground instances C' of N such that $C' \prec C$, and N^C be the set of ground instances C' of N such that $C' \preceq C$. Now suppose L is the maximal literal in C and let R be a (ground) rewrite system. Then we write R_C for the set of rules $l \approx r$ from R such that $l \approx r \prec L$, and R^C for the rules $l \approx r \preceq L$. (This notation is consistent with that of definition 1.)

For any rewrite system R, set of closures N, and ground closures D and C, let us say that D *follows from the R-reduced part of N_C* if there exist ground instances $D_1, \ldots, D_k$ of N such that (i) $C \succ D_i$, for $1 \leq i \leq k$, (ii) if D is reduced with respect to R then so is each D_i, and (iii) if each D_i is true in R^{D_i}, then D is true in R_D.

Definition 2 We call a ground closure D *redundant* with respect to N, if for any convergent ground rewrite system R for which D is reduced, D follows from the R-reduced part of N_D. Whenever the set R is obvious, we will also say that D is redundant with respect to $D_1, \ldots, D_k$, referring to the D_i that imply D in the sense made precise above.

For convenience in this subsection, temporarily call a position *selected* via the given selection rule S in a ground instance $A \cdot \rho\tau$ of a closure $A \cdot \rho$ from a given set N if it is selected in $A \cdot \rho$, and analogously for the redex ordering $\prec_{\mathcal{R}}$.

Definition 3 A ground instance of an equality resolution or equality factoring inference from N is *redundant* with respect to N if, for any convergent ground R for which the premise C is order-irreducible at substitution and selected positions, the conclusion D follows from the R-reduced part of N_C.

A ground instance

$$\frac{C' \vee s \approx t \quad C}{D}$$

(where p is the redex position in C) of a paramodulation inference is said to be *redundant* with respect to N if either some premise is redundant with respect to N, or else D follows from the R-reduced part of N_C, for any convergent ground rewriting system R containing the rule $s \approx t$ and for which the positions in P are order-irreducible, where P is the union of the substitution positions in both premises, the selected positions in the left premise, and the selected positions $q \prec_{\mathcal{R}} p$ in the second premise.

Finally, a closure (or an inference) is called redundant if all its ground instances are redundant.[8]

Note that an equality resolution or equality factoring inference is redundant by this definition if its premise is redundant. This characterization of which closures and inferences are unnecessary in constructing a model for a set of closures provides us with a characterization of which closures and inferences are unnecessary in searching for a refutation for an inconsistent set of closures. This provides a framework for designing useful syntactic criteria for elimination and simplification of closures.

The completeness results in this paper depend on the properties of sets of closures in which all non-redundant inferences have been performed.

Definition 4 We say that a set of closures N is *saturated* if every inference from N is redundant with respect to N.

Saturated sets have special properties which provide for the completeness of our inference rules.

Lemma 6 *Let N be a saturated set of closures which does not contain the empty clause, R be a rewrite system constructed from N according to definition 1, and let $C = \tilde{C} \cdot \rho\tau$ be an R-reduced ground instance of a closure $\tilde{C} \cdot \rho$ in N. Then*

(i) C is true in R_C if (i.1) C is redundant, or (i.2) C is order-reducible by R_C at a selected position, or (i.3) some negative equation in C is selected;

(ii) If C is false in R_C then it must be a productive clause of the form $C = C' \vee s \approx t$ (where $s \approx t$ is the equation produced), such that C' is false in R, and

(iii) C is true in R and in R_D, for every $D \succ C$.

Proof. First of all we note that (iii) follows from (i) and (ii), by corollary 1. Therefore we prove only the first two cases, proceeding by induction on the clause ordering $\succ$. Suppose N is saturated and does not contain the empty clause, and assume that properties (i) — (iii) hold for all reduced ground instances D of N with $C \succ D$. We consider each subcase in turn.

(i.1) Suppose that C is redundant with respect to R-reduced ground instances D_i, $1 \leq i \leq k$, of N. By the induction hypothesis we know that

[8] For a clause or inference to be redundant crucially depends on the choice of the ordering $\succ$ and the vocabulary Ω with respect to which ground instances are considered. In cases where we have to emphasize this dependency we will speak of redundancy with respect to $\succ$ and Ω.

each D_i is true in R_C (and hence in R^{D_i}), from which we may conclude that C is true in R_C.

Let us therefore assume that C is not redundant. We proceed by contradiction by assuming that C is false in R_C. In this case we show that there exists a ground instance of an inference from N with C and (in the case of paramodulation) a productive clause D with $C \succ D$, as premises; we then show that the conclusion B of the ground inference must be a reduced closure which is false in R_C. Using the induction hypothesis for (i) and (ii) we may infer that D is not redundant. Because N is saturated the inference is redundant; but since neither premise is redundant, then B follows from the R-reduced part of N_C, so there exist reduced ground instances $D_1 \dots D_k$ of N which are smaller than C. By the induction hypothesis, the D_i are true in R_C, and so B is true in R_C, a contradiction.

Therefore in what follows we need only provide for the existence of the reduced conclusion B false in R_C from premises C and, in the case of a paramodulation, a productive clause $D \prec C$. Note in this argument that B need not be a ground instance of N and we do not apply the induction hypothesis to B.

(i.2) Suppose C is order-reducible at a selected position p by a rule $s \approx t$ in R, but is false in R_C. Furthermore, assume that p is the least such reducible selected position with respect to the redex ordering $\prec_{\mathcal{R}}$, and that $s \approx t$ is produced by a ground clause $D = D' \vee s \approx t$. As $s \approx t$ is in R_C, $C \succ D$. Using the induction hypothesis for (i) and (ii) and lemma 4 we may infer that D is represented by a reduced ground instance $\tilde{D} \cdot \rho\tau$ (of a closure $\tilde{D} \cdot \rho$ from N)[9] which is order-irreducible at selected positions, and has no negative selected equations; furthermore, D' is false in R, and $s \approx t$ is true in R.

We distinguish two cases, depending on whether p occurs in the negative or positive literals of C.

In the first case, if $C = C' \vee u[s]_p \not\approx v$, then $u \not\approx v \succ D$ because $u \not\approx v \succ s \approx t$ and $s \approx t$ is maximal in D. If t is selected, then it is irreducible by R, and since $u \approx v \in R_C$ and so $u[t] \Downarrow_{R_C} v$, then either $u = s$ and $v \succeq t$, or $u \succ s$, with the result that $u \approx v \succeq s \approx t$ as required. Thus there exists a ground instance

$$\frac{D' \vee s \approx t \quad C' \vee u[s]_p \not\approx v}{C' \vee D' \vee u[t] \not\approx v}$$

[9] Again, for simplicity, we use ρ and τ for the substitutions in both closures, since these are variable disjoint.

of an inference satisfying all the ordering and other conditions for paramodulation; let $B \cdot \theta$ denote the conclusion of the ground inference and $\tilde{B} \cdot \theta'$ be the result of some number of variable abstractions applied to this conclusion. Note that we have $B\theta \prec C$ because $s \succ t$ and $u \not\approx v \succ D$. Now we know, using the induction hypothesis for (ii), that D' is false in R and in R_C. Also $u[t] \approx v$ is in R_C, as both $u \approx v$ and $s \approx t$ are. Finally, C' is false in R_C, with the result that B is false in R_C. This provides for the necessary contradiction as mentioned above, as long as we can show that $\tilde{B} \cdot \theta'$ is reduced.

First we verify that $B \cdot \theta$ is reduced. Consider how this closure is derived from $D = \tilde{D} \cdot \rho\tau$ and $C = \tilde{C} \cdot \rho\tau$. The fact that the premises are reduced implies that every equation in $C' \vee D'$ is reduced. It remains to show that $u[t] \not\approx v$ is reduced. Let x be a (closure) variable in t. If $l \approx r \in R$ reduces $x\theta$, then $s \succ t \succeq l$. Hence, $s \approx t \succ l \approx r$, and $l \approx r$ would also order-reduce $x\rho\tau$ in the occurrence $s \approx t$ in D, which is a contradiction. If x is a (closure) variable in $u[t] \approx v$ but not in t, then any equation smaller than the occurrence of $u[t] \not\approx v$ and reducing $x\theta$ would also reduce $x\rho\tau$ in the occurrence of $u[s] \not\approx v$ in C. As $u[s] \approx v \succ u[t] \approx v$ we again obtain a contradiction.

From this it is easy to see that $\tilde{B} \cdot \theta'$ is reduced. This is because all selected terms in D, and all selected terms less than p with respect to $\prec_{\mathcal{R}}$ in C, are reduced by hypothesis, and because any other term abstracted must be relatively reduced to some other substitution term by the definition of variable abstraction.[10]

This derives the contradiction in the case that position p occurs in a negative literal. The case where p occurs in a positive literal is completely analogous. The only significant difference is that we know that $u \approx v \succ D$ because either $u \succ s$ or $u = s$ and $v \succ t$ (since if $u = s$ and $v = t$ then C would not be false in R_C), so $u \approx v \succ s \approx t$. The remainder of the argument is almost identical.

(i.3) Next, consider the case where some negative equation in C is selected. By the previous cases, we may assume that C is not redundant and is order-irreducible at selected positions. Again we assume that the clause is false in R_C, which means that all negative equations in C must be true in R_C. Thus C must be in the form $C' \vee s \not\approx s$, where $s \not\approx s$ is the selected

[10] Observe that this inference is a ground instance of an inference from N, and hence variable abstraction is applied only to the conclusion of this general inference, and not at the ground level.

equation, since it is irreducible by R. Consider the ground instance

$$\frac{C' \vee s \not\approx s}{C'}$$

of an equality resolution inference from N (the reader may easily check that the conditions for such an inference are satisfied). Clearly $C' \prec C$ and C is false in R_C. The proof that C' is reduced is trivial, since any term at a variable position in C' also occurs at a variable position in C, and, as with the previous case, any variable abstractions would not change the fact that the conclusion is reduced.

(ii) Suppose that C is false in R_C. From case (i), we may assume that C is a non-redundant instance which is order-irreducible at selected positions by R, and which contains no negative selected equations. We also know that C is not the empty clause. Therefore C must be in the form $C' \vee s \approx t$, where $s \approx t$ is maximal, $s \succ t$ (since C can not be a tautology), and thus s is selected. We distinguish two subcases, depending on whether $s \approx t$ is strictly maximal in C.

If it is, then the clause is reductive, and since s is irreducible in R_C then the clause produces $s \approx t$. Since C' is false in R_C, the only thing that remains is to show that the positive equations in C' remain false in R. Now suppose to the contrary that $C' = C'' \vee u \approx v$, where $u = v$ is true in R. Since C' is false in R_C, we have $u \approx v \in I \setminus R_C$, which is only possible if $s = u$ and $t \Downarrow_{R_C} v$, with $t \succ v$. Consider the ground instance

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

of an equality factoring inference, where B is the conclusion. Note that t can not be selected, since then it would be normalized, violating the fact that $t \Downarrow_{R_C} v$ with $t \succ v$. Hence condition (iv) for equality factoring is satisfied. The other conditions are easily checked. Now, since $s \approx v \succ t \not\approx v$, then $C \succ B$, but since C and the literal $t \not\approx v$ are false in R_C, so is B. The only thing which remains in order to derive the contradiction as in case (i) is to show that B is reduced. This depends on the observation that any (closure) variable x in t or v in the conclusion occurs also in the premise in one of the strictly larger equations $s \approx t$ or $s \approx v$. Subsequent variable abstractions, again, would keep the conclusion reduced.

Now suppose that $s \approx t$ is not strictly maximal in C. Then $C' = C'' \vee s \approx t$, and we proceed almost exactly as in the previous paragraph. The only difference is that we proceed with the assumption that $t = v$; therefore if t is in the form $t' \cdot \rho\tau$ and v in the form $v' \cdot \rho\tau$, then $t\rho$ and $v\rho$ must be unifiable (satisfying condition (iv) for equality factoring). This concludes case (ii) and the lemma. $\square$

This result allows us to show that the process of saturating a set of closures of the form $C \cdot \mathrm{id}$ will produce the empty closure iff the set is inconsistent. (In the following section we will discuss methods for saturation.)

Theorem 1 *Let K be a set of clauses and let N be a saturated set of closures such that $C \cdot \mathrm{id}$ is in N for any clause C in K and such that any closure in N follows from K. Then K is consistent if and only if N does not contain the empty clause. In the latter case, R is a model of K and N.*

Proof. If N contains the empty clause, K is inconsistent. On the other hand, if N does not contain the empty clause, R is a model of any R-reduced instance of N, as was shown in lemma 6. Now let $C\rho$ be a ground instance of K. We define a substitution τ by $x\tau = t_x$, where t_x is the normal form of $x\rho$ by R. Then $C \cdot \tau$ is a reduced ground instance of the closure $C \cdot \mathrm{id}$ in N. Therefore $C\tau$, and hence $C\rho$, is true in R. $\square$

5 Theorem Proving in the Presence of Deletion Rules

We now discuss the completeness of methods for saturating a set of closures in which we may delete superfluous closures. The central notion of this section, that of a *fair theorem proving derivation*, is introduced in Subsection 5.2. The basic idea here is that at each step in the process of refutational theorem proving, we can either add a consequence of the existing set of clauses, or delete a subsumed or a redundant closure. After this definition, we present a number of specific applications of redundancy, such as simplification and blocking. However, our definition of redundancy does not explain the special case of subsumption by a clause with the same number of literals[11] and so we present the notion of subsumption first, in Subsection 5.1, and incorporate it more essentially into the definition of a fair theorem proving derivation.

[11] It would be possible to modify the definition of redundancy to accomodate this special case, at the cost of some additional complexity, and so we have chosen to deal with the problem outside the framework of redundancy.

Before we present these results, it will be convenient to have a set of purely syntactic sufficient conditions for the notion of redundancy for closures. For that purpose the notion of "relative reducibility" of closures is significant.

Definition 5 A ground closure $C \cdot \sigma$ is *reduced relative* to another ground closure $D \cdot \theta$ if for any R, $C \cdot \sigma$ is R-reduced whenever $D \cdot \theta$ is.

For example, $P(g\boxed{b})$ is reduced relative to $P(\boxed{fb})$. For non-ground closures, this notion must be extended slightly for the contexts in which we use it.

Definition 6 A position q in a literal L is reduced relative to a position p in a literal L' [closure C] *modulo* η if for any R and for any ground instance $L'\tau$ [$C\tau$], $L\eta\tau$ is reduced at q whenever $L'\tau$ [$C\tau$] is reduced at p. A closure $D \cdot \sigma$ is called reduced relative to $C \cdot \theta$ *modulo* η if for any R and for any R-reduced ground instance $C \cdot \theta\tau$, $D \cdot \sigma\eta\tau$ is R-reduced at all positions at which a variable $x \in dom(\sigma)$ occurs.

For example, the position of gfy in $P(gfy)$ is reduced relative to the position of gfy (modulo the identity substitution), but not relative to the position of fy, in $Q(gfy)$. The closure $P(\boxed{fy})$ is reduced relative to $Q(\boxed{fgx})$ modulo $\{y \mapsto gx\}$ but not modulo $\{y \mapsto gc\}$.

The notion of "relatively reduced" is rather strong, as it requires this property to hold for *any* rewrite system, but fortunately there are simpler sufficient conditions. The essential idea is that relative reducibility can be assured in all but pathological cases by checking that the respective substitution terms in the first closure are a subset of the substitution terms in the second. For instance, a closure $D \cdot \sigma$ is reduced relative to $C \cdot \theta$ modulo η if for every position p where a variable $x \in dom(\sigma)$ occurs in a literal M in D, there exists a variable y occurring in some literal L in C, such that $x\sigma\eta$ is a subterm of $y\theta$ and either $L\theta \succeq M\sigma$ or $L\theta$ is negative.

The only pathologies involve substitution terms at the maximal side of a positive equation. For example, $P(\boxed{fx})$ is not reduced relative to $\boxed{fx} \approx c$. For supposing $b \succ c$, the ground instance $\boxed{fb} \approx c$ is reduced with respect to $fb \approx b$, but $P(\boxed{fb})$ is not.

One issue concerning closures which are reduced relative to each other needs to be clarified at this point. If $C \cdot \sigma$ and $D \cdot \theta$ are two closures such that $C\sigma$ and $D\theta$ are identical up to variable renaming, and each is reduced relative to the other, then they are said to be *identical upto renaming and* *under reducibility*. For example, $Q(\boxed{a}) \vee P(a,x)$ and $Q(\mathrm{a}) \vee P(\boxed{a}, y)$ are identical in this sense. We will see later that in our inference system such closures need not be distinguished.

We now present a set of sufficient conditions for redundancy which are of practical significance for theorem proving.

Lemma 7 *Let $D, D_1, \ldots, D_k$ be closures from a set N, and $\eta_1, \ldots, \eta_k$ be substitutions such that*

1. *For each i, $D_i\eta_i \prec D$,*
2. *For each i, D_i is reduced relative to D modulo η_i, and*
3. *For any ground instance $D\tau$ of D, $D\tau$ is a consequence of $D_1\eta_1\tau, \ldots, D_k\eta_k\tau$.*

Then D is redundant in N.

Proof. Let R be a convergent ground rewriting system, and $D\tau$ be an R-reduced ground instance of D. Note that each variable in each $D_i\eta_i$ occurs in D, since $D_i\eta_i \prec D$. Thus each $D_i\eta_i\tau$ is ground. Now, for any ground substitution $\theta = \{x_j \mapsto t_j\}_{1\leq j\leq n}$, temporarily define $\theta\!\downarrow$ as $\{x_i \mapsto t'_j\}$, where t'_j is the normal form of t_j with respect to the rewrite system $R_{D\tau}$. We claim that the set

$$D_1(\eta_1\tau)\!\downarrow, \ldots, D_k(\eta_k\tau)\!\downarrow$$

satisfies conditions (i) – (iii) in the definition of redundancy.

First, for each i, clearly $D_i(\eta_i\tau)\!\downarrow \preceq D_i\eta_i\tau \prec D\tau$, so condition (i) is satisfied. Now, suppose $D_i = \tilde{D}_i \cdot \sigma_i$. Because $D\tau$ is R-reduced, we must show that $\tilde{D}_i \cdot \sigma_i(\eta_i\tau)\!\downarrow$ is R-reduced. (This is not trivial, because $(\eta_i\tau)\!\downarrow$ being normalized does not of itself imply that $x\sigma_i(\eta_i\tau)\!\downarrow$ is normalized.) Now, for any occurrence of a variable x in $\tilde{D}_i$ there are two cases. If $x \notin dom(\sigma_i)$, then $x\sigma_i(\eta_i\tau)\!\downarrow = x(\eta_i\tau)\!\downarrow$ is R-normalized by definition. Otherwise, if $x \in dom(\sigma_i)$, then since D_i is reduced relative to D modulo η_i, we know $x\sigma_i(\eta_i\tau)$ is order-irreducible by R, and so any proper subterm is in R-normal form (since it can not be at the top of a maximal side of a positive equation). We conclude that $x\sigma_i(\eta_i\tau)\!\downarrow = x\sigma_i\eta_i\tau$, and so $x\sigma_i(\eta_i\tau)\!\downarrow$ is order-irreducible. Thus $D_i(\eta_i\tau)\!\downarrow$ is R-reduced. This verifies condition (ii).

Now, for (iii) we first observe that the sequence of lemmas culminating in corollary 1 are true not only for models constructed according to our definition, but for arbitrary ground convergent rewrite systems. Thus, assume that each $D_i(\eta_i\tau)\downarrow$ is true in $R^{(D(\eta_i\tau)\downarrow)}$; then it must be true in $R_{D\tau}$ as well, by the extension of Corollary 1. But then by (3) above, $D\tau$ is true in $R_{D\tau}$. $\Box$

This set of three conditions can be used to prove the completeness of the next two deletion rules we discuss.

5.1 Basic Subsumption

First we present the form of subsumption which is used in the basic setting. A closure C is a *basic subsumer* of a closure D if there exists a substitution η such that $C\eta$ is a submultiset of D, and C is reduced relative to D modulo η; it is a *proper basic subsumer* if D is not a basic subsumer of C in turn. Basic subsumption reduces to standard subsumption in the case of closures with identity substitutions.

Note that non-proper basic subsumers are identical upto renaming and under reducibility, as defined in Subsection 2.6. A technical feature of proper basic subsumption which will be used later is the following.

Lemma 8 *The relation "is a proper basic subsumer of" is well-founded and transitive.*

Proof. The only difficulty is in proving well-foundedness. We map each closure C to a complexity measure $< P, M >$, where P is the number of non-variable positions in C, and M is the multiset of integers $\{k_1, \dots k_m\}$, where $var(C) = \{x_1, \dots, x_m\}$ and each x_i occurs k_i times. The lexicographic combination of $>$ and $>_{mul}$ is well-founded on such pairs. If $C\eta = D$, then C has a strictly smaller complexity, since either C has fewer literals than D (reducing the first component), η maps some variable in C to a non-variable term (reducing the first component), or else C and D have the same number of literals and η maps two variables in C to some single variable in D (reducing the second). $\Box$

When a closure C is a basic subsumer of a closure D, then D may be deleted from the set of closures. The technical justification for this deletion rule is that subsumed clauses are unnecessary in constructing a model for a set of clauses. In most cases, this is because of redundancy.

Lemma 9 *Let C be a basic subsumer of D, where C contains fewer literals than D. Then D is redundant with respect to C.*

Proof. We simply observe that C with its associated η fits the criteria mentioned in lemma 7. $\square$

The other case of subsumption we will deal with in the next subsection. A natural question at this point is what to do when one clause subsumes another in the standard sense but not the the basic sense (i.e., is not relatively reduced). That we can not naively delete such subsumed clauses in the basic setting is shown by the next example.

Example 1

$$\begin{array}{lcl} \neg P(x,y) & \vee & P(x,b) \\ \neg P(a,b) & & \\ & & a \approx c \\ & & P(c,b) \end{array}$$

Suppose we use a lexicographic path ordering based on the precedence $P \succ Q \succ a \succ b \succ c$. If we resolve the first two clauses, we obtain the clause $\neg P(\boxed{a}, y)$. Since this new clause subsumes (in the standard sense) the second clause, we might suppose that the latter clause can be deleted. However, if we do so, the reader may verify that there is no refutation. Note that this would not be a legal subsumption step in the basic setting, unless we retracted $\neg P(\boxed{a}, y)$ to $\neg P(a, y)$ before performing the deletion.

If we have a subsumer in the standard, but not the basic sense, then we may retract the subsumer in such a way that it is reduced relative to the closure subsumed. Since we wish to keep as much of the closure in the substitution part as possible, this means retracting just enough of the substitution part of the subsumer so as to satisfy the condition of relative reducibility. We now discuss a simple deterministic way of achieving this, by giving a sufficient condition for "relatively reduced" which essentially requires that the substitution part of one closure can be overlapped in a very straight-forward way onto the substitution part of another.

Definition 7 Let $s \cdot \sigma$ and $t \cdot \theta$ be closures of terms and let us temporarily define P as the set of positions in t where non-variable subterms occur. Also, suppose that $dom(\sigma) \subseteq var(s)$. We say that $s \cdot \sigma$ is *η-dominated* by $t \cdot \theta$, for some substitution η, written $s \cdot \sigma \sqsubseteq_\eta t \cdot \theta$, iff $s\sigma\eta = t\theta$ and for each $x \in dom(\sigma)$, if x occurs in s at position p, then $p \notin P$. For equations, we say that $(s \approx t) \cdot \sigma \sqsubseteq_\eta (u \approx v) \cdot \theta$ iff either $s \cdot \sigma \sqsubseteq_\eta u \cdot \theta$ and $t \cdot \sigma \sqsubseteq_\eta v \cdot \theta$, or if

$s{\cdot}\sigma \sqsubseteq_\eta v{\cdot}\theta$ and $t{\cdot}\sigma \sqsubseteq_\eta u{\cdot}\theta$. For negated equations the definition is analogous. For closures of multisets of literals, we have $C_1 \cdot \theta_1 \sqsubseteq_\eta C_2 \cdot \theta_2$ iff there exists an injection φ from $C_1 \cdot \theta_1$ into $C_2 \cdot \theta_2$ such that if $\varphi(L_1 \cdot \theta_1) = L_2 \cdot \theta_2$, then $L_1 \cdot \theta_1 \sqsubseteq_\eta L_2 \cdot \theta_2$. For closures of clauses, we have $(\Delta \to \Gamma) \cdot \sigma \sqsubseteq_\eta (\Pi \to \Theta) \cdot \rho$ iff $\Delta \cdot \sigma \sqsubseteq_\eta \Pi \cdot \rho$ and $\Gamma \cdot \sigma \sqsubseteq_\eta \Theta \cdot \rho$.

We write $\Phi \sqsubseteq \Psi$ to indicate that there exists some η such that $\Phi \sqsubseteq_\eta \Psi$ (in the case of closures, Φ is in fact a basic subsumer of Ψ).

Note that this relation is not closed under substitution, since for example $Px \cdot id \sqsubseteq Pa \cdot id$ but $Px \cdot [x \mapsto a] \not\sqsubseteq Pa$.

The basic idea of the relation $\sqsubseteq_\eta$ is that all terms in the closure substitution on the left side must overlap directly onto the right side inside the closure substitution. Clearly this is a sufficient condition for one literal, or one closure, to be reduced relative to another modulo η. But it is not necessary, since for example $P(\boxed{a}, b)$ is reduced relative to $P(b, \boxed{a})$, but $P(\boxed{a}, b) \not\sqsubseteq_\eta P(b, \boxed{a})$. However, for subsumption and simplification (to be presented below) it is a relatively simple condition to check, and provides for a simple method for forming the minimal retract when the condition fails. Roughly, if $L' \cdot \sigma\rho = L \cdot \theta$ but $L' \cdot \sigma \not\sqsubseteq_\rho L \cdot \theta$, then we can take the union U of the set of non-variable skeleton positions in $L' \cdot \sigma$ and in $L \cdot \theta$, and form the retract $L'' \cdot \sigma'$ of $L' \cdot \sigma$ by instantiating the positions in U (equivalently, this can be thought of as taking the intersection of substitution positions).

We will return to another application of this test for relative reducibility when we consider simplification in a later subsection.

5.2 Fair Saturation Methods

Complete methods for theorem proving amount to procedures for saturating a set of clauses with respect to a given set of inference rules.

Definition 8 A (finite or countably infinite) sequence $N_0, N_1, N_2, \ldots$ of sets of closures is called a *theorem proving derivation* if the substitution part of every closure in N_0 is empty, and if each set N_{i+1} can be obtained from N_i by adding a clause which is a consequence of N_i or by deletion of a redundant or a subsumed clause. A closure C is said to be *persisting* if there exists some j such that for every $k \geq j$, there exists a closure C' in N_k which is identical with C upto renaming and under reducibility.[12] The set of all persisting closures, denoted N_∞, is called the *limit* of the derivation.

[12] Naturally, C and C' may be the same closure.

A theorem proving derivation is called *fair* if N_∞ is saturated.

This means that a fair derivation can be constructed, for instance, by systematically adding conclusions of non-redundant inferences from persisting closures. We can also apply various deletion rules during this process, as redundant closures and inferences stay redundant through the course of a theorem proving derivation.

Lemma 10 *(i) If $N \subseteq N'$, then any closure [inference] which is redundant with respect to N is also redundant with respect to N'.*

(ii) If $N \subseteq N'$ and all closures in $N' \setminus N$ are redundant with respect to N', then any closure [inference] which is redundant with respect to N' is also redundant with respect to N.

Proof. It is sufficient to consider only the case of ground instances of closures and inferences. For (i) the result is trivial for both closures and inferences, since $N \subseteq N'$. Thus consider (ii) in the case of closures. Let a ground instance D be redundant with respect to N', suppose an arbitrary R is given, and assume that we choose the set $D_1, \ldots, D_k$ as the minimal such with respect to $\prec_{mul}$. If we can prove that no member of this set is itself redundant wrt N', then D is redundant with respect to N. Thus, suppose some D_i is redundant with respect to a set $E_1, \ldots, E_n$ of ground instances of N'. But then we can show that D is redundant with respect to

$$D_1, \ldots, D_{i-1}, E_1, \ldots, E_n, D_{i+1}, \ldots, D_k.$$

Clearly conditions (i) and (ii) in definition 3 are still satisfied; and if each E_i is true in R^{E_i}, then D_i is true in R_{D_i}, and thus (by Corollary 1) D_i is true in R^{D_i}, and so each of the D_i, $1 \leq i \leq k$, is true in R^{D_i}, and the original condition (iii) applies; thus our original set was not minimal, a contradiction.

Next we consider part (ii) of the lemma in the case of inferences. The case of redundancy on account of redundant premises is covered by the previous paragraph. Thus, consider an inference from N' with premises $C_1 \ldots C_n$ and conclusion C, which is redundant in N' by virtue of a set $\{D_1 \ldots D_k\}$ of instances of N' with the properties specified in the definition of a redundant inference. As above, we may assume that no D_i is redundant, which means that $\{D_1 \ldots D_k\} \subseteq N$ and the inference is redundant in N. $\Box$

This shows a fundamental property of redundancy: redundancy is preserved if additional closures are added or if redundant closures are deleted.

Redundancy is a syntactic means of determining if a clause is unnecessary in the process of saturating a set, and has as special cases most of the common deletion rules used in theorem provers. There are some instances of deletion rules which can not be proved complete using the notion of redundancy we employ, for example (as mentioned above) the special case of subsumption by a closure with the same number of equations. However, such closures are unnecessary in constructing a model for a set of closures. The main completeness result of the paper may now be given.

Theorem 2 *Let $N_0, N_1, N_2, \ldots$ be a fair theorem proving derivation. If $\bigcup_j N_j$ does not contain the empty closure, then N_0 is consistent.*

Proof. Since N_∞ is saturated and does not contain the empty closure, by lemma 6 we can construct a rewrite system R with an associated interpretation for the set. It remains to be shown that this yields a model of $\bigcup_j N_j$, from which we conclude that N_0 is consistent. It suffices to show that R is a model of any ground instance C of $\bigcup_j N_j \setminus N_\infty$. There are two cases.

Suppose such a C is not redundant in $\bigcup_j N_j$. Then by lemma 10 (i) it can not be a ground instance of a closure which was redundant at some finite stage N_i. The only remaining possibility is that C is subsumed by some ground instance C' of $\bigcup_j N_j$ with the same number of literals. Now, by lemma 8, we may assume that C' is the minimal such under the proper subsumption relation, and so there is no C'' which properly subsumes C'. Since C' can not be redundant in $\bigcup_j N_j$ (or else so would be C, since C' is reduced relative to C), then it must be in N_∞ and hence C' and C are true in R.

Next, suppose C is redundant with respect to $\bigcup_j N_j$. By lemma 10 (ii) it is redundant with respect to R-reduced ground instances $D_1 \ldots D_k$ of $\bigcup_j N_j$ which are not themselves redundant. But then by lemma 6 and the previous paragraph, each D_i is true in R, and so C is true in R. This concludes the proof. $\Box$

5.3 Basic Simplification

Simplification techniques in our calculus can be designed and justified using the sufficient conditions for redundancy developed in a previous subsection. The main problem, as with subsumption, is to insure that the relative reducibility criterion holds, however, we also wish to preserve as much of the constraint of the closure as possible during the simplification process, and this causes some additional complications. We present two versions of simplification, the first a very general rule using variable abstraction, and a second version based the sufficient condition $\sqsubseteq$ which avoids variable abstraction.

Let $D[l']_p \cdot \theta$ be a closure with l' a non-variable skeleton term, which is order-reducible at p by an instance $l\sigma\rho \approx r\sigma\rho$ of a closure equation $(l \approx r) \cdot \sigma$ from N which is reduced relative to $D[l']_p \cdot \theta$ modulo ρ and such that $l\sigma\rho \succ r\sigma\rho$. Then we can *basic simplify* this closure into the form

$$D[r\rho] \cdot \sigma\rho\theta.$$

Then we perform variable abstraction of this new closure wrt the old closure. (Note that by the assumption of variable disjointness for closures, and by the idempotence of the substitutions, $\sigma\rho\theta = \theta + \sigma\rho$.)

The simplified version of the closure D is added to the set and the original can then be deleted because (as we show below) it is then redundant. The main difficulty is in insuring that the new closure and the simplifier are reduced relative to the original D, modulo the matching substitution. If the simplifier does not satisfy this condition, then we can form a retract which does. Naturally, we would wish to retract as few positions in the simplifier as possible. An additional complication is that is that some variables in l may not be bound by σ, and if these also occur in r, then we must instantiate them when r is inserted into the simplified closure to insure that it is reduced relative to the old one. For example, we can not simplify $P(f(a)) \cdot id$ by $(f(x) \approx g(x)) \cdot id$ to obtain $P(g(x)) \cdot \{x \mapsto a\}$, but must instantiate x by the matching substitution to obtain $P(g(a))$. The information about substitution positions in the original closure which is lost during this process can then be recovered by variable abstraction.

An example may perhaps clarify this rule. Suppose a closure

$$Pf(g\boxed{a}, h(\boxed{hb})) \quad = \quad Pf(gw, hw') \cdot \{w \mapsto a, w' \mapsto hb\}$$

is to be simplified by a closure

$$f(x, \boxed{hhz}) \approx k(x, \boxed{hhz}) \quad = \quad f(x, y) \approx k(x, y) \cdot \{y \mapsto hhz\}.$$

Then the matching substitution is $\rho = \{x \mapsto ga, z \mapsto b\}$, however we must take a retract of the rule in order to perform the simplification. For example, we may form the new rule

$$f(x, h(\boxed{hz})) \approx k(x, h(\boxed{hz})) \quad = \quad f(x, hv) \approx k(x, hv) \cdot \{v \mapsto hz\}.$$

Now we have relative reducibility modulo ρ and may simplify the literal to

$$Pk(ga, h(\boxed{hb})) \quad = \quad Pk(ga, hv) \cdot \{v \mapsto hb\}$$

according to our rule (we have surpressed useless bindings).

However, note that we have lost the fact that a is considered to be irreducible by the original closure. Thus we could abstract out the a to obtain

$$Pk(g\boxed{a}, h(\boxed{hb})) \quad = \quad Pk(gv', hv) \cdot \{v' \mapsto a, v \mapsto hb\}.$$

Our first version of simplification, in combination with variable abstraction, is the most general form of simplification rule in our calculus.

However, if the condition $\sqsubseteq$ is used to insure relative reducibility, then certain details of the general method above become more concrete. The idea here is similar to the case of subsumption: we must insure that the term in the simplifier is dominated by the term in the clause being matched, and could form the retract of the simplifier by taking the intersection of the non-variable substitution positions in $l' \cdot \theta$ and $l \cdot \sigma$. In the same spirit, we would also need to form a retract in which $var(r) \subseteq var(l)$.

In fact, in the example above, we formed the retract in this way to obtain relative reducibility via the condition that

$$f(x, hv) \cdot \{v \mapsto hz\} \sqsubseteq_\rho f(gw, hw') \cdot \{w \mapsto a, w' \mapsto hb\}.$$

In this framework we can express the variable abstraction process directly in the simplification rule. Let us suppose we add the conditions that $var(r) \subseteq var(l)$ and $l \cdot \sigma \sqsubseteq_\rho l' \cdot \theta$ in our formulation of simplification, so that ρ is a matcher of $l\sigma$ onto $l'\theta$. Let $p_1, \ldots, p_n$ be the positions of all occurrences of variables in $l\sigma$. The matcher ρ binds these variables to subterms of $l'\theta$. The only problematic variables are those x such that for every occurrence of x in $l\sigma$ at position q, q is a non-variable postition in l'; for all other variables y, some $y\rho$ occurs at a substitution position in $l' \cdot \theta$ and hence can be preserved in the substitution part of the simplified term. For problematic x, we can not assume that the whole term $x\rho$ is reduced relative to the clause being simplified. Our original version of simplification solved this in brute force fashion by simply instantiating each such term by replacing the redex by $r\rho$ (the problematic variables are all in $dom(\rho)$). However, as demonstrated above, we lose information about the portions of such problematic $x\rho$ which are known to be reduced by virtue of overlapping substitution positions in $l' \cdot \theta$. To calculate the "minimal instantiatiation" $r\rho'$, for each variable $x \in var(l\sigma)$ occurring at positions $q_1, \ldots, q_m$, if any q_j occurs at a substitution position of $t \cdot \theta$, then define $x\rho' = x$; otherwise, let $x\rho'$ be the most specific generalization (see Huet 1980) of the terms $l'/q_1, \ldots, l'/q_m$. Thus the problematic variables are exactly $dom(\rho')$. Since $x\rho = (l'/q_1)\theta = \ldots = (l'/q_m)\theta$, then for each $x \in dom(\rho')$, $x\rho'$ contains only variables already occurring in l', and $x\rho'\theta = x\rho$. Now, for each problematic variable x, the substitution postitions in $x\rho' \cdot \theta$ are relatively reduced to the closure being simplified, since they are a part of θ. Therefore we reformulate the simplification rule so that the simplified clause is of the form $D[r\rho'] \cdot \sigma\rho\theta$ and do not perform variable abstraction. This implementation of basic simplification reduces to standard simplification when $\sigma = \theta = id$ (cf. also the complete version of simplification used in basic narrowing as in Nutt, Rety, & Smolka 1989, where $\sigma = id$).

For example, in simplifying $f(h(x,b), h(a,y)) \cdot \{x \mapsto a, y \mapsto b\}$ by $(f(z,z) \approx z) \cdot id$, with the matching substitution $\rho = \{z \mapsto h(a,b)\}$, our original rule would give us a reduction to $h(a,b) \cdot id$ before variable abstraction produces $h(x', y') \cdot \{x' \mapsto a, y' \mapsto b\}$. We may perform this reduction directly by taking the most specific generalization $h(x,y)$ of $h(x,b)$ and $h(a,y)$,and forming $\rho' = \{z \mapsto h(x,y)\}$ (note that $z\rho'\theta = h(a,b) = z\rho$), we would simplify the term to $h(x,y) \cdot \{x \mapsto a, y \mapsto b\}$.

Note that in the context of "eager" application of the variable abstraction rule to conclusions of inferences, the terms $l'/q_1, \ldots, l'/q_m$ would all be identical and the use of most specific generalization would not be necessary. In fact, most of the fussy details above are only necessary to avoid a special requirement that variable abstraction be so used.

To sum up, when using the sufficient condition $\sqsubseteq$ for relative reducibility, we can preserve as much of the original constraint on the simplified closure as possible by instantiating the replacement term r by just as much of the matcher ρ as overlaps only on the skeleton of the clause being simplified when the match from $l\sigma$ onto l' is calculated, the portion overlapping θ being already "safe" for abstraction. The point here is to preserve as much information about the frontier of a closure as possible throughout the simplification process.

The justification for deleting a clause after a simplified version has been constructed is again that it is redundant. The proof is again a routine verification of the conditions in lemma 7 to show that the original closure is redundant in the context of the simplifier and the newly simplified closure.

Lemma 11 *Let* $C = (l \approx r) \cdot \sigma$, $D' = D[l'] \cdot \theta$, $D'' = D[r\rho] \cdot \sigma\rho\theta$, *and* $D''' = D[r\rho'] \cdot \sigma\rho\theta$ *be as above. Then* D' *is redundant with respect to* C *and* D'', *and with respect to* C *and* D'''.

As with subsumption, the standard notion of simplification (i.e., where relative reducibility does not hold) is incomplete in the basic setting, as the following example shows.

Example 2

$$\begin{array}{ll} P(f(x)) & \vee\ f(x) \approx b \\ \neg P(f(a)) & \\ & a \approx c \\ f(c) \not\approx b & \end{array}$$

We assume a lexicographic path ordering based on the precedence $P \succ f \succ a \succ b \succ c$, and suppose the selection rule simulates superposition, as discussed in section 3. Let us assume that saturation begins with resolving the first onto the second clause. This produces a closure, $f(\boxed{a}) \approx b$, which we use to simplify the second clause, to obtain

$$\begin{array}{ll} P(f(x)) & \vee\ f(x) \approx b \\ \neg P(b) & \\ & a \approx c \\ f(c) \not\approx b & \\ & f(\boxed{a}) \approx b \end{array}$$

From hereon it is impossible to derive the empty clause by basic superposition, as the calculus does not admit a superposition of $a \approx c$ into $f(\boxed{a}) \approx b$.

5.4 Basic Blocking

The sufficient conditions for redundancy given in lemma 7 are fairly general, but do not provide for all deletion rules which we would like to implement. Two other rules we will discuss are essentially a kind of tautology deletion: if we know that for every model represented by a convergent rewrite system R, every R-reduced instance of a closure C is true in R, then C can be deleted, since it is redundant by our definition. The first rule, blocking, occurs when there are no R-reduced instances and also can be extended to a rule for blocking inferences.

The main idea in this subsection is that the generation of simplifiers in the process of saturating a set of closures allows us to reason to some degree about the model constructed for the "final" saturated set. Briefly, if a simplifier $l \approx r$ appears, and $l\sigma \succ r\sigma$ for some σ, then any occurrence of $l\sigma$ in a clause will represent the location of a term which is reducible with respect to the R constructed from the saturated set. This means that if $l\sigma$ occurs at a substitution position, then the closure is not reduced and hence not necessary in the construction upon which our completeness result rests.

Definition 9 Let us call a instance $l\sigma\rho \approx r\sigma\rho$ of a closure $(l \approx r) \cdot \sigma$ from N a *basic simplifier instance of* N if $l\sigma\rho \succ r\sigma\rho$. A closure $C \cdot \theta$ is *blocked* with respect to a set of closures N if it is order-reducible at a substitution position by a basic simplifier instance of N which is reduced relative to $C \cdot \theta$.

Note that relative reducibility always holds in this case if $var(r) \subseteq var(l)$. Blocked closures can always be deleted from a set.

Lemma 12 *Blocked closures are redundant.*

Proof. Suppose $C \cdot \theta$ is order-reducible at a substitution position in a literal $L \cdot \theta$ by $(l \approx r) \cdot \sigma\rho$. For notational simplicity let us assume that the closures are ground (otherwise we would consider ground instances via some ground substitution τ). Thus suppose $C \cdot \theta$ is reduced with respect to some R; we claim that $(l \approx r) \cdot \sigma\rho$ satisfies conditions (i)—(iii) in the definition of redundancy. Clearly (i) and (ii) hold. Now suppose $(l \approx r) \cdot \sigma\rho$ is true by virtue of equations in R no larger than itself; then the term $l\sigma\rho$ is reducible by an equation in R no bigger than $l\sigma\rho \approx r\sigma\rho$. But then again $L \cdot \theta$ would be reducible at a substitution position by a smaller equation. In either case this implies that $C \cdot \theta$ was not R-reduced, a contradiction. Thus (iii) must hold trivially. $\Box$

In blocking, the left side of a rule is trivially reduced relative to the substitution term which it matches modulo the matching substitution ρ; thus we need only verify that the right side is relatively reduced. A simple way to ensure this, as mentioned above, is to verify that $var(r) \subseteq var(l)$ or form a relatively reduced retract. An example which shows that the relative reducibility of the right side is necessary in blocking may be framed as follows (a similar example could be constructed for simplification).

Example 3

$$\begin{array}{rcl} & & P(a,b) \\ \neg P(x,y) & \vee & Q(x,f(y)) \\ \neg Q(a,x) & \vee & a \approx x \\ & & f(b) \approx c \\ \neg Q(a,c) & & \end{array}$$

Suppose an ordering based on the precedence $P \succ Q \succ R \succ a \succ f \succ b \succ c$. If we resolve the first two clauses, we obtain the clause $Q(\boxed{a}, f(\boxed{b}))$. Then if we resolve this new clause with the third clause, we obtain the clause $a \approx \boxed{f(b)}$, which blocks $Q(\boxed{a} \vee f(\boxed{b}))$. Since the variables of the right hand side of the blocking equation are not in the left hand side, the equation should be instantiated. If we do not perform the instantiation, $f(b) \approx c$ blocks $a \approx \boxed{f(b)}$. Therefore, both of the new clauses can be deleted; we are left with the original set of clauses, and because of fairness, no more inferences need be performed. We have not found a refutation, although the original set was unsatisfiable.

Note that an inference

$$\frac{C_1 \cdot \rho \ \cdots \ C_n \cdot \rho}{C \cdot \theta}$$

is redundant by definition if one of the closures $C_1 \cdot \theta \ \cdots \ C_n \cdot \theta$ is blocked. It is possible in addition to show that certain additional inferences can be blocked during the saturation of a set of clauses; this is essentially a generalization of the technique of blocking due to Slagle (1974) (see also Lankford 1975 and Hsiang & Rusinowich 1991)

Definition 10 An equality resolution or equality factoring inference with premise $C \cdot \rho$ and conclusion $D \cdot \theta$ is *blocked* in N if $C \cdot \theta$ is blocked or if $C \cdot \theta$ is order-reducible at a selected position by a basic simplifier instance $l \approx r \cdot \sigma\rho$ of N which is reduced relative to the substitution and selected positions in $C \cdot \theta$.

Consider a paramodulation inference

$$\frac{(C' \vee s \approx t) \cdot \rho \quad C[s']_p \cdot \rho}{D \cdot \theta}$$

(where p is the redex position), let $C_1 = (C' \vee s \approx t) \cdot \theta$, and let $C_2 = C[s']_p \cdot \theta$. Define P as the union of the selected positions in C_1, the selected positions $q \prec_{\mathcal{R}} p$ in C_2, and the substitution positions in both these closures. The inference is *blocked* in N if

(i) it is order-reducible at a position in P in C_1 or C_2 by a basic simplifier instance as above of N which is relatively reduced to the positions P, or

(ii) it is order-reducible in C_2 by the instance $s\theta \approx t\theta$, at either a selected position $q \prec_{\mathcal{R}} p$ or at a substitution position.

Note that case (i) includes the possibility that either C_1 or C_2 is blocked (as a closure). The reader should compare this definition with the definition of a redundant inference given previously. As explained above, the fundamental idea here is that the equations used to do reduction can be assumed to be true in the model R, and hence indicate the presence of reducible terms. Note that for a simplifier, we can use an arbitrary instance, whereas in part (ii), we must use the instance $s\theta \approx t\theta$ generated by the paramodulation inference (i.e., it can not be further instantiated). This is because any instance of a positive unit clause must be true, but we do not know which instances (if any) of $s\theta \approx t\theta$ are true.

Lemma 13 *Blocked inferences are redundant.*

Proof. The case where the premises are blocked is trivial by the definition of a redundant inference. For the other cases it if sufficient to consider ground inferences. Thus, consider an equality resolution or equality factoring inference with conclusion $D \cdot \theta$ and with a premise $C \cdot \theta$ which is order-reducible at a selected position by a basic simplifier ground instance $l \approx r$ reduced relative to the selected and substitution positions in the premise. Then for any R for which $C \cdot \theta$ is reduced at substitution and selected positions, we can show that $l \approx r$ satisfies conditions (i)—(iii) in definition 3. The only difference from the similar argument in lemma 12 is that we consider selected positions in addition to substitution positions.

Now consider a ground paramodulation inference with premises $(C_1 \vee s \approx t) \cdot \theta$ and $C_2[s']_p \cdot \theta$ and conclusion $D \cdot \theta$, and which is reducible at a position in P as specified in case (i) by basic simplifier ground instance $l \approx r$ which is reduced relative to the positions P. Again for any R we can show that $l \approx r$ satisfies conditions (i)—(iii) in definition 3, by considering reducibility at substitution and selected positions. If the inference is order-reducible by $s\theta \approx t\theta$ at a position as specified in case two, the argument is identical, except that we consider a rewrite system R containing $s\theta \approx t\theta$. $\Box$

Under certain very natural conditions, selection rules can be used to precalculate which clauses will cause inferences to be blocked, and so the work in actually constructing the inferences and checking these conditions can be saved. For example, if the selection rule is invariant under substitution, then a clause which is simplifiable at a selected position q will form a blocked inference whenever it is either the first premise or else the second premise of a paramodulation applied at a position bigger than q.

In addition, it will sometimes be possible to perform simpler checks for blocking when the set of simplifiers has special properties. For instance, if a set of simplifiers fully defines a function symbol f in the sense that every ground term containing f is reducible by a basic simplifier instance, then it is sufficient simply to check for the existence of f in substitution and selected terms when blocking.

5.5 Basic Tautology Deletion

Another deletion rule which can be shown to be correct using the notion of redundancy is tautology deletion. For example, a simple kind of tautology in paramodulation has the form $C \vee \neg A \vee A$ or the form $C \vee s \approx s$, and can be shown to be redundant with respect to the empty set of closures. This is because a clause which is always true in any model is unnecessary in the construction of models. Thus *any* tautology can be deleted. In our setting, in addition, it is possible to define another kind of tautology by virtue of the fact that we represent models by convergent rewrite systems and require closures to be reduced (at the ground level) in our completeness proof. This implies that for any convergent rewrite system R, an R-reduced ground equation of the form $(x \approx s) \cdot \sigma$, where $x\sigma \succ s\sigma$, must always be false with respect to R, since any rewrite proof between the two sides must reduce $x\sigma$. When such an equation occurs negatively in a clause C, then C must be true with respect to R.

Definition 11 A clause of the form $(C \vee x \not\approx s) \cdot \sigma$ is a *basic tautology* if $x\sigma \succ s\sigma$.

A routine verification of the conditions for redundancy, in the case where the set $\{D_1, \ldots, D_k\}$ is empty, gives us the following result.

Lemma 14 *Basic tautologies are redundant in any set N.*

It is also possible to do a similar check during the construction of an inference

$$\frac{C_1 \cdot \rho \ \cdots \ C_n \cdot \rho}{C \cdot \theta}$$

on closures. If any of $C_1 \cdot \theta \ \cdots \ C_n \cdot \theta$ are tautologies or basic tautologies, then the inference is redundant and need not be performed.

6 Basic Completion

We next look at the relationship between the Knuth/Bendix completion method and saturation up to redundancy. One question is under what circumstances a saturated set of equations is convergent (and not just ground convergent). In this section we consider only positive unit clauses, which for simplicity can be thought of as equations. (Since we will only reason about saturated sets below, we need not consider closures, but only the clauses represented by them.)

By a *basic completion procedure* we mean any procedure that accepts as input a set of equations E and a reduction ordering $\succ$ and generates a fair theorem proving derivation from E in which all deduction steps are by basic paramodulation and all deletion steps are by basic simplification, basic subsumption, or blocking. We have shown that the interpretation generated from the limit E_∞ of a fair derivation is a model of E which can be represented by a convergent ground rewrite system R consisting of certain ground instances of E_∞. Thus, the set of all orientable ground instances of E (that is, the set of all instances $s\sigma \approx t\sigma$, for which $s\sigma \succ t\sigma$) is convergent on ground terms. In this sense, saturation of a (finite or recursively enumerable) set of equations up to redundancy under the basic strategy may be thought of as a basic variant of the ordered completion procedure.

An interesting situation arises when all equations in E_∞ are orientable with respect to $\succ$. We will show that in that case, E_∞ is actually convergent on *all* terms. Let $\mathcal{F}$ be the given set of function symbols and $\mathcal{V}$ be the given set of variables. We first introduce a set of new constants $\mathcal{C}$, such that a bijection $\iota : \mathcal{V} \to \mathcal{C}$ exists. Furthermore, let $\kappa : \mathcal{C} \to \mathcal{F}$ be the function that maps each constant in $\mathcal{C}$ to the same minimal (with respect to $\succ$) constant in $\mathcal{F}$.

The reduction ordering $\succ$ can be extended to an ordering $\succ_\kappa$ on $\mathcal{T}(\mathcal{F} \cup \mathcal{C}, \mathcal{V})$ as follows (Bachmair, Dershowitz, and Plaisted 1989): $s \succ_\kappa t$ if and only if either $\kappa(s) \succ \kappa(t)$ or else $\kappa(s) = \kappa(t)$ and $s \succ_{lpo} t$. (Here $\succ_{lpo}$ denotes a lexicographic path ordering based on a total well-founded precedence relation on $\mathcal{F} \cup \mathcal{C}$ and the mapping κ is extended from $\mathcal{C}$ to $\mathcal{T}(\mathcal{F} \cup \mathcal{C})$ in the usual way.) Note that $\succ_\kappa$ is indeed a reduction ordering that extends $\succ$ and moreover is total on the set of ground terms $\mathcal{T}(\mathcal{F} \cup \mathcal{C})$ (cf. Bachmair 1991).

Lemma 15 *Let $\succ$ be a reduction ordering that is total on $\mathcal{T}(\mathcal{F})$ and E be a set of equations between terms in $\mathcal{T}(\mathcal{F}, \mathcal{V})$, such that $s \succ t$, for all equations $s \approx t$ in E. Then, for all terms u and v in $\mathcal{T}(\mathcal{F}, \mathcal{V})$ with $\iota(u) \Rightarrow_{E^{\succ_\kappa}} \iota(v)$ we have $u \Rightarrow_{E^{\succ}} v$.*

Proof. Suppose u and v are terms of the form $u[s\sigma]$ and $u[t\sigma]$, respectively, where $s \approx t$ is an equation in E and $\iota(u) \succ_\kappa \iota(v)$. We have either $s \succ t$ or $t \succ s$, so that $\kappa(\iota(u)) \neq \kappa(\iota(v))$. This implies $\kappa(\iota(u)) \succ \kappa(\iota(v))$, from which we may infer $s \succ t$ and hence $u \succ v$. $\Box$

We have the following result.[13]

Theorem 3 *Let $\succ$ be a reduction ordering that is total on $\mathcal{T}(\mathcal{F})$. Let E be a set of equations between terms in $\mathcal{T}(\mathcal{F}, \mathcal{V})$ and E_∞ be the limit constructed by a basic completion procedure for inputs E and $\succ$. If $s \succ t$, for all equations $s \approx t$ in E_∞, then E_∞ is a convergent rewrite system on $\mathcal{T}(\mathcal{F}, \mathcal{V})$.*

Proof. First observe that any fair derivation from E with respect to $\succ$ (over the set of ground terms $\mathcal{T}(\mathcal{F})$) can also be interpreted as a fair derivation from E with respect to $\succ_\kappa$ (over the set of ground terms $\mathcal{T}(\mathcal{F} \cup \mathcal{C})$). The limit E_∞ of the derivation is thus convergent on all ground terms in $\mathcal{T}(\mathcal{F} \cup \mathcal{C})$. We claim that it is also convergent on $\mathcal{T}(\mathcal{F}, \mathcal{V})$.

If u and v are terms in $\mathcal{T}(\mathcal{F}, \mathcal{V})$, such that $u \Leftrightarrow^*_{E_\infty} v$, then $\iota(u) \Leftrightarrow^*_{E_\infty} \iota(v)$ and, by ground convergence, $\iota(u) \Downarrow_{E_\infty^{\succ_\kappa}} \iota(v)$. By the above lemma we get $u \Downarrow_{E_\infty^{\succ}} v$, which completes the proof. $\Box$

The substitution positions in the rewrite systems produced by basic completion have no significance when such systems are used for reduction, however it is interesting that when these systems are used for basic narrowing (see below), substitution positions can be added to the positions at which narrowing is forbidden. This can be easily seen by recasting narrowing problems of the form $R \models \exists(s \approx t)$? in the form of a refutation of the set $R \cup \{s \not\approx t\}$ using the inference systems presented here; see also Chabin, Anantharaman & Rety 1993.

[13] We remind the reader of the caveat expressed in the footnote to definition 3.

7 Summary

In this paper we have defined a framework for paramodulation (and completion) which depends on a reduction ordering, a selection function, and a redex ordering to restrict inferences along several dimensions. The "basic" strategy forbids inferences into substitution positions. Ordering restrictions work both at the level of clauses, at the level of literals, and at the level of terms to restrict inferences. (We remark here that it is possible to refine the notion of selection in a way analogous to the notion of a "complete set of positions" in Fribourg (1989). Essentially, we only need to select positions which include some redex at the ground level so that we may provide for an inference in the completeness proof. The definition of selection given in this paper is a very general one which assumes no special information about the clauses. With more information, for example in the presence of additional constraints on clauses, it may be possible to restrict selection.) Selection is particularly significant in defining restrictions on inference positions in negative literals, whereas orderings are more significant on positive literals. Finally, redex orderings on selected positions define reducibility criteria on positions in clauses. These results can be thought of as defining the frontier between the explored and unexplored parts of the clause and for controlling the application of inference rules in the unexplored regions. In addition to the standard inference rules, variable abstraction can be performed to extend the basic restriction on closures, and a variety of deletion rules which implement a very general notion of redundancy have been presented.

The basic strategy was introduced explicitly—as far as we know—for the first time in Russia by Degtyarev (1979), who sketches a basic strategy for paramodulation, but we do not have any detailed information about his calculus. It was introduced in the West in a more comprehensive way by Hullot (1980), and further studied by Nutt, Rety & Smolka (1989). This latter paper shows that the basic strategy conflicts to some degree with simplification, and a method for dealing with this was described. In addition, various of the techniques described in this paper, such as selection, blocking non-reduced closures, and variable abstraction, were described in a comprehensive framework. Redex orderings are a more general form of the Left-to-Right Basic Narrowing rule of Herold (1986) and Bosco et al. (1987) (see also Bockmayr et al. 1992). The current paper can thus be thought of as an extension and development of techniques discovered first in the narrowing framework to the full first-order calculus in a refutational setting.

D. Plaisted has remarked to us that some features of Brand's modification method (Brand 1975) are reminiscent of the basic strategy, and the theorem prover described by Nie & Plaisted (1990), which uses a similar transformation, also avoids paramodulation into substitution terms. A critical pair criterion similar to the basic strategy is described in Smith & Plaisted (1988). McCune (1990) has conjectured that it is never necessary to perform paramodulation inside Skolem functions. Indeed, this refinement is a special case of basic paramodulation, and hence the conjecture is a corollary of our results. More generally, (basic) paramodulation inferences need never be applied to proper subterms of function symbols, such as Skolem symbols, that occur in the input clauses with just variables as arguments.

R. Nieuwenhuis and A. Rubio have also independently developed an inference system for completion and for refutational theorem proving based on basic superposition and proved completeness in the context of deletion rules such as subsumption and simplification (Nieuwenhuis & Rubio 1992a). In addition, they have developed a comprehensive framework for ordering constraints in combination with equational constraints (essentially the same as our closure substitutions) and analysed the role of initial constraints and problems with deletion in this framework (Nieuwenhuis & Rubio 1992b). Although we have not stressed it here, this paper can be seen as a contribution to the theory of constrained rewriting and deduction (see Kirchner, Kirchner & Rusinowich 1990). The current project grew out of a lemma necessary in the proof of Snyder & Lynch (1991), and was presented in a preliminary form at the *4th Unification Workshop* in Barbizon, France, without deletion or blocking rules, and using a very different style of proof. The current paper is a long version of the abstract presented at the Eleventh Conference on Automated Deduction (Bachmair et al. 1992).

Our results, in addition to providing a means of making paramodulation theorem provers (and related systems, such as completion procedures) more efficient, show that substitutions, which are produced initially as most general unifiers which calculate the intersection of ground instances of universally quantified clauses, in fact play *only* this role in theorem proving, in the sense that they need not be subject to equational inferences themselves. We view these results as a robust answer to the question posed by L. Wos and cited in the introduction in the following sense. Essentially, our results depend on the fact that terms in clauses can be forbidden for paramodulation inferences when, at the ground level, they represent irreducible terms in the construction of the model described in Section 4. The user specifying additional forbidden terms in the original set of clauses—which would be more in the spirit of set of support—seems to require that we can prove that these clauses are *reduced* to start with; since in general it is difficult or impossible to know what models could be constructed for a set of clauses being saturated (except in a limited sense when simplifiers arise), it seems that the results presented here contain the strongest possible such restrictions.

Acknowledgments. We wish to thank Dennis Kfoury and Steve Homer of Boston University for graciously providing funds for the third author during the academic years 1990–92. We would also like to thank Michael Rusinowich, David Plaisted, Pierre Lescanne, Deepak Kapur, H.-J. Ohlbach, Robert Nieuwenhuis, Albert Rubio, Zino Benaissa, Nachum Dershowitz, and the anonymous referees for helpful discussions on the ideas presented here. Thanks also to Jean-Pierre Jouannaud for his interest in this work.

References

[1] L. Bachmair and H. Ganzinger. On Restrictions of Ordered Paramodulation with Simplification. In *Proc. 10th Int. Conf. on Automated Deduction*, Lect. Notes in Comput. Sci., vol. 449, pp. 427–441, Berlin, 1990. Springer-Verlag.

[2] L. Bachmair and H. Ganzinger. Rewrite-based Equational Theorem Proving with Selection and Simplification. To appear in *Journal of Logic and Computation* (1994).

[3] L. Bachmair, H. Ganzinger, C. Lynch, and W. Snyder. Basic Paramodulation and Superposition. In *Proc. 11th Int. Conf. on Automated Deduction*, Lect. Notes in Artificial Intelligence, vol. 607, pp. 462–476, Berlin, 1992. Springer-Verlag.

[4] L. Bachmair. *Canonical Equational Proofs*. Birkhauser Boston, Inc., Boston MA (1991).

[5] A. Bockmayr, S. Krischer, and A. Werner. An Optimal Narrowing Strategy for General Canonical Systems. In *Proc. of CTRS*, M. Rusinowich and J.L. Remy (Eds.), LNCS vol. 250, pp. 483–497, Berlin, 1992.

[6] P.G. Bosco, E. Giovannetti, and C. Moiso. Refined Stategies for Semantic Unification. In *Proc. of TAPSOFT'87*, LNCS, vol. 250, H. Ehrig et al. (Eds.), pp. 276-290, Berlin, 1987.

[7] D. Brand. Proving Theorems with the Modification Method. *SIAM Journal of Computing 4*:4 (1975) pp. 412–430.

[8] J. Chabin, S. Anantharaman, & P. Rety. E-Unification via Constrained Rewriting. In *Proceedings of Seventh Workshop on Unification*, Boston University, F. Baader & W. Snyder (Organizers), June 1993.

[9] A. Degtyarev. The Monotonic Paramodulation Strategy. In *Proc. 5th All-Union Conference on Mathematical Logic*. Novosibirsk (1979). (In Russian.)

[10] N. Eisinger A Note on the Completeness of Resolution without Self-resolution. *Information Processing Letters 31* (1989) pp. 323-326.

[11] M. Fay. First-order Unification in an Equational Theory. In *Proc. 4th Workshop on Automated Deduction*, Austin, Texas (1979).

[12] L. Fribourg. A Strong Restriction of the Inductive Completion Procedure. *J. Symbolic Computation 8* (1989) 253–276.

[13] A. Herold. Narrowing Techniques Applied to Idempotent Unification. SEKI-Report SR-86-16, Univ. Kaiserslautern (1986).

[14] J. Hsiang and M. Rusinowich. Proving Refutational Completeness of Theorem Proving Strategies: The Transfinite Semantic Tree Method. *J. ACM 38* (1991) 559–587.

[15] G. Huet Confluent Reductions: Abstract Properties and Applications to Term Rewriting Systems. J. ACM 27 (1980) 797–821.

[16] J.-M. Hullot. Canonical Forms and Unification. In *Proc. 5th Int. Conf. on Automated Deduction*, Lect. Notes in Comput. Sci., vol. 87, pp. 318–334, Berlin, 1980. Springer-Verlag.

[17] S. Kaplan. Positive/negative Conditional Rewriting. In *Conditional Term Rewriting Systems*, Lect. Notes in Comput. Sci., vol. 308, pp. 129–143, Berlin, 1988. Springer-Verlag.

[18] C. Kirchner, H. Kirchner and M. Rusinowich. Deduction with Symbolic Constraints. *Revue Francaise d'Intelligence Artificielle Vol 4, no. 3* (1990) pp. 9-52.

[19] Krischer, S., and A. Bockmayr. Detecting Redundant Narrowing Derivations by the LSE-SL Reducibility Test. In *Proc. 4th Int. Conf. on Rewriting Techniques and Applications*, Lect. Notes in Comput. Sci., vol. 488, pp. 74–85, Berlin, 1991. Springer-Verlag.

[20] D. Lankford. Canonical Inference. *Tech. Rep. ATP-32*, Dept. of Math. and Comp. Sci., Univ. of Texas, Austin, TX (1975).

[21] W. McCune. Skolem Functions and Equality in Automated Deduction. In *Proc. 8th Nat. Conf. on AI*, MIT Press, 1990, pp. 246–251.

[22] X. Nie and D. Plaisted. A cOmplete Semantic Back-chaining Proof System. In *Proc. 10th Int. Conf. on Automated Deduction*, Lect. Notes in Comput. Sci., vol. 449, pp. 16–27, Berlin, 1990. Springer-Verlag.

[23] R. Nieuwenhuis and A. Rubio. Basic Superposition is Complete. In *Proc. European Symposium on Programming*, Rennes, France (1992).

[24] R. Nieuwenhuis and A. Rubio. Theorem Proving with Ordering Constrained Clauses. In *Proc. 11th Int. Conf. on Automated Deduction*, Lect. Notes in Artificial Intelligence, vol. 607, pp. 477–491, Berlin, 1992. Springer-Verlag.

[25] W. Nutt, P. Réty, and G. Smolka. Basic Narrowing Revisited. *J. Symbolic Computation 7* (1989) 295–317. Reprinted in *Unification*, C. Kirchner (ed.), Academic Press, London (1990).

[26] J. Pais and G. Peterson. Using Forcing to Prove Completeness of Resolution and Paramodulation. *J. Symbolic Computation 11* (1991) pp.3-19.

[27] G. Peterson. A Technique for Establishing Completeness Results in Theorem Proving with Equality. *SIAM Journal of Computing 12* (1983) pp. 82–100.

[28] G.A. Robinson and L. T. Wos. Paramodulation and Theorem Proving in First-order Theories with Equality. In B. Meltzer and D. Michie, editors, *Machine Intelligence 4* pp. 133–150. American Elsevier, New York, 1969.

[29] J. Slagle. Automated Theorem Proving with Simplifiers, Commutativity, and Associativity. *J. ACM 21* (1974) pp. 622–642.

[30] M. Smith and D. Plaisted. Term-rewriting Techniques for Logic Programming I: Completion. Report TR88-019, Department of Computer Science, Univ. North Carolina (1988).

[31] W. Snyder and C. Lynch. Goal Directed Strategies for Paramodulation. In *Proc. 4th Int. Conf. on Rewriting Techniques and Applications*, Lect. Notes in Comput. Sci., vol. 488, pp. 150–161, Berlin, 1991. Springer-Verlag.

[32] W. Snyder. *A Proof Theory for General Unification*. Birkhauser Boston, Inc., Boston, MA (1991).

[33] L. T. Wos, G. A. Robinson, D. F. Carson, and L. Shalla. The Concept of Demodulation in Theorem Proving. *Journal of the ACM*, Vol. 14, pp. 698–709, 1967.

[34] L. Wos. *Automated Reasoning: 33 Basic Research Problems*. Prentice Hall, Englewood Cliffs, New Jersey (1988).

[35] H. Zhang. *Reduction, Superposition, and Induction: Automated Reasoning in an Equational Logic*. Ph.D. Thesis, Rensselaer Polytechnic Institute (1988).