Chapter 2

Resolution Theorem Proving

Leo Bachmair

Harald Ganzinger

Second readers: David McAllester and Christopher Lynch.

Contents

HANDBOOK OF AUTOMATED REASONING
Edited by Alan Robinson and Andrei Voronkov

1. Introduction

Saturation-based theorem proving in its modern form was invented by Robinson [1965*b*] when he introduced the resolution calculus, the essence of which can be described by two inference rules:

(Binary) Resolution

$$\frac{C \vee A \quad D \vee \neg B}{(C \vee D)\sigma}$$

where σ is the most general unifier of the atomic formulas A and B;

(Positive) Factoring

$$\frac{C \vee A \vee B}{(C \vee A)\sigma}$$

where σ is the most general unifier of the atomic formulas A and B.

Resolution is a refutationally complete theorem proving method: a contradiction (i.e., the empty clause) can be deduced from any unsatisfiable set of clauses. The search for a contradiction proceeds by saturating the given clause set, that is, systematically (and exhaustively) applying all inference rules.

Resolution on ground clauses is a version of the cut rule restricted to atomic formulas, whereas factoring is an instance of contraction.[1] In fact, the refutational completeness of resolution can be derived from the completeness of the (propositional) sequent calculus; and Herbrand's theorem, which states that for any unsatisfiable set of non-ground clauses there is a finite set of ground instances that is propositionally unsatisfiable, establishes a link between propositional clauses and general clauses with variables. A key in the "lifting" argument is the existence (and uniqueness) of a most general unifier for any two unifiable atoms or terms.

But resolution is not primarily a method for deciding the unsatisfiability of propositional formulas (the procedure by Davis and Putnam [1960], for instance, is better suited for that purpose). Its main advantage over other early theorem proving methods, such as Gilmore's algorithm [1960], is that unification, as a selection mechanism for inferences, provides an effective way of interleaving the two processes: (i) the identification of suitable (ground) instances of clauses and (ii) a demonstration of their unsatisfiability.

In this chapter we describe the theoretical concepts and results that form the basis of state-of-the art automated theorem provers based on resolution and refinements thereof. After presenting some preliminary material in Section 2, we explain, in Section 3, the main ingredients of resolution calculi — orderings and selection — and the main theoretical concepts — candidate models and reduction of counterexamples — which we use for obtaining completeness results for these calculi. In

[1]Some textbooks (e.g., Gallier [1986]) take a different perspective and present resolution as a macro inference in the cut-free sequent calculus. However, when the clauses to be refuted are viewed as additional non-logical axioms, cuts can not be eliminated, but may be restricted to analytic cuts—the resolution inferences.

Section 4 we describe a general framework for modeling theorem proving processes that involve both simplification and search. This formalism is applied to an extended example of a resolution-based theorem prover with simplification in Section 4.3. In Sections 5, 6, and 7 we present, and refine, a general resolution calculus for general clauses, i.e., clauses which are (disjunctive) multisets of arbitrary quantifier-free formulae. This calculus serves mainly as a theoretical concept from which most of the major resolution-based calculi (binary resolution, positive resolution, semantic resolution, hyper-resolution, non-clausal resolution, theory resolution, the inverse method, Boolean ring-based methods) can be derived as special cases. Viewing the specialized calculi as different instances of one general inference system also sheds light on their mutual relationships.

For simplicity the presentation of most inference systems is initially given for variable-free formulas only. In Section 9 we briefly discuss techniques for lifting our results to the general level of formulas with variables.

We will also indicate how the theory of resolution can be applied to obtain refinements of tableau-based theorem proving methods by arguing, in Section 8, that the notion of a "closed tableau" can be generalized to that of a "saturated tableau" in which all paths are saturated, up to redundancy, by ordered resolution. In Section 10 we discuss the role of resolution-based methods, not only for refutational theorem proving, but also as a tool for analyzing and compiling presentations of logical theories. It will be briefly explained how saturation may help in automatically generating decision procedures for a theory so that certain complexity bounds for the entailment problem for the theory can be guaranteed. We also show that saturation may be used as a tool for generating variants of resolution calculi that are specifically tailored to certain theories such as orderings or congruences. This provides new insights on how to compute with large, but structured theories.

A distinctive feature of our presentation is that we not only place local restrictions on resolution inferences, but via an abstract notion of redundancy also provide a framework in which global restrictions on the proof search can be expressed and justified by means of simplification and elimination. This allows us to answer questions about the compatibility, or incompatibility, of such techniques as tautology elimination, subsumption, normal form transformation, or reduction with various types of resolution calculi.

2. Preliminaries

2.1. Formulas and Clauses

We consider quantifier-free first-order formulas built from variables, function symbols, predicate symbols and logical connectives. We will deal with the logical symbols $\top$ (verum), $\bot$ (falsum), $\neg$ (negation), $\vee$ (disjunction), $\wedge$ (conjunction), $\supset$ (implication), $\oplus$ (exclusive disjunction), and $\equiv$ (equivalence), though our results apply to other connectives as well. A *term* is either a variable or an expression $f(t_1, \ldots, t_n)$, where f is an n-ary function symbol and $t_1, \ldots, t_n$ are terms.

An *atomic formula* (or *atom*) is an expression $P(t_1, \ldots, t_n)$, where P is a predicate symbol of arity n and $t_1, \ldots, t_n$ are terms. A predicate symbol of arity 0 is called a *propositional constant*. A *literal* is an expression A (a *positive* literal) or $\neg A$ (a *negative* literal), where A is an atomic formula. Two literals A and $\neg A$ are said to be *complementary*.

Calculi for automated deduction are often described in terms of constructs that represent formulas, but abstract from certain non-essential aspects of the syntax or encode additional structural information. For example, multiple disjunctions or conjunctions may be conveniently represented as sequences (or multisets), due to the associativity (and commutativity) property of these connectives.

A *multiset* over a set S is a function Σ from S to the natural numbers. Intuitively, $\Sigma(x)$ specifies the number of occurrences of x in Σ. We say that x is an *element* of Σ if $\Sigma(x) > 0$. A set may be thought of as a multiset Σ for which $\Sigma(x)$ is 0 or 1, for all x. A multiset Σ is *finite* if $\Sigma(x) = 0$ for all but finitely many x. The *union* and *intersection* of multisets are defined by the identities $\Sigma_1 \cup \Sigma_2(x) = \Sigma_1(x) + \Sigma_2(x)$ and $\Sigma_1 \cap \Sigma_2(x) = \min(\Sigma_1(x), \Sigma_2(x))$. If Σ is a multiset and S a set, we write $\Sigma \subseteq S$ to indicate that every element of (the multiset) Σ is an element of (the set) S, and use $\Sigma \setminus S$ to denote the multiset Σ' for which $\Sigma'(x) = 0$ for any x in S, and $\Sigma'(x) = \Sigma(x)$, otherwise. We often use sequences or set-like notation to denote multisets and write, for instance, Σ, Δ instead of $\Sigma \cup \Delta$, or Σ, A instead of $\Sigma \cup \{A\}$. For example, by $\neg A, B, B$ we denote the multiset Σ over formulas for which $\Sigma(\neg A) = 1$, $\Sigma(B) = 2$, and $\Sigma(F) = 0$, for all other formulas F.

A finite multiset of formulas may either be interpreted as the disjunction or as the conjunction of its elements. We will interpret multisets as disjunctions and speak of *general clauses*. The empty multiset represents the constant $\bot$. If $(F_1, \ldots, F_n)$ is a general clause, we write $\neg(F_1, \ldots, F_n)$ to denote the formula $\neg F_1 \wedge \ldots \wedge \neg F_n$. If all the elements in a general clause are literals, it is called a *standard clause*. Standard clauses are usually written as disjunctions, $L_1 \vee L_2 \vee \ldots \vee L_n$. We use calligraphic letters $\mathcal{C}$, $\mathcal{D}$, and $\mathcal{G}$ to denote general clauses, and capital letters C and D to denote standard clauses.

On rare occasions we will interpret multisets as conjunctions, in which case we speak of *dual* (general or standard) clauses. Note that the empty dual clause represents the constant $\top$.

We write $E[E']_p$ to denote the expression that is obtained from replacing the subexpression at position p in E by E'.[2] Thus, $E[E']_p$ is an expression that contains E' as a subexpression (at position p). The position p may be omitted if it is clear from the context. Sometimes we use p to denote a set of positions in E, in which case replacement is meant to take place at all the positions in p simultaneously. We also write $E(E')$ to indicate that E contains at least one occurrence of E' as a subexpression. By $E[E'/E'']$ or, if E' is known from the context, simply $E(E'')$, we denote the result of simultaneously replacing *all* occurrences of E' in E by E''.

Variable-free expressions are called *ground* or *closed*. When we wish to emphasize that an expression may contain variables we also speak of a *first-order expression*.

[2]Positions may be represented in Dewey decimal notation, as sequences of non-negative integers.

2.2. Herbrand Interpretations

A (*Herbrand*) *interpretation* is a set of ground atoms. A ground atom A is said to be *true* in an (Herbrand) interpretation I if $A \in I$, and *false* otherwise. The logical connectives are interpreted in the usual way. The constant $\top$ is true in all interpretations, whereas $\bot$ is false in all interpretations. A conjunction $A \wedge B$ is true in I if both A and B are true in I; a disjunction $A \vee B$ is true if at least one of A and B is true; etc. The truth value of a formula depends only on the truth values assigned to its atomic formulas. A general clause $(F_1, \ldots, F_n)$ is true in I if at least one of the formulas F_i is true in I (whereas a dual clause $(F_1, \ldots, F_n)$ is true in I if all of the formulas F_i are true in I).

An interpretation I is called a *model* of an expression E if E is true in I; and a model of a set of expressions N, if it is a model of all expressions in N. An expression or a set of expressions is called *satisfiable*, or *consistent*, if it has a model; and *unsatisfiable*, or *inconsistent*, otherwise. An expression that is true in all interpretations is said to be *valid*, or a *tautology*. We also say that E' is a *logical consequence* of E, or *logically follows* from E, or that E *logically implies* E' (written $E \models E'$), if E' is true in all models of E. Two expressions E and E'' are said to be *(logically) equivalent*, written $E \simeq E''$, if, and only if, they have the same truth value in each interpretation (i.e., are either both true or both false). By a *contradiction* we mean an inconsistent expression that contains only the constants $\top$ and $\bot$ and logical connectives, but no function or predicate symbols. For example, $\bot$ and $\top \supset \bot$ are contradictions, whereas $A \wedge \neg A$ is inconsistent, but not a contradiction in this sense.

2.3. Rewrite Systems

Rewrite systems are a basic tool for describing a variety of theorem proving techniques. We use the letters $\alpha, \beta, \ldots$ to denote variables ranging over formulas. Syntactically, these "meta-variables" are treated like propositional constants.

A *substitution* is a mapping defined on variables, where variables denoting terms are mapped to terms and variables denoting formulas, to formulas. 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* (i.e., contains no variables), we speak of a *ground instance* of E. 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.

An *equivalence (relation)* is a reflexive, transitive, symmetric binary relation. For example, logical equivalence is indeed an equivalence relation. A binary relation $\Rightarrow$ on expressions with variables is called a *rewrite relation* if $E' \Rightarrow E''$ implies $E[E'] \Rightarrow E[E'']$, for all expressions E, E' and E''. If $\Rightarrow$ is a binary relation, we denote by $\Rightarrow^+$ its transitive closure; by $\Rightarrow^*$ its transitive-reflexive closure; by $\Leftrightarrow$ its symmetric closure; and by $\Leftrightarrow^*$ its transitive-reflexive-symmetric closure.

A *rewrite system* is a binary relation on expressions with variables, the elements of which are called *rewrite rules* and written $E \Rightarrow E'$. (We occasionally speak of a *two-way rewrite rule*, and write $E \Leftrightarrow E'$, if a rewrite system contains both $E \Rightarrow E'$ and $E' \Rightarrow E$.) If R is a rewrite system, we denote by $\Rightarrow_R$ the smallest rewrite relation that contains all instances $E\sigma \Rightarrow E'\sigma$ of rules in R. We say that E can be *rewritten to* E' by R, if $E \Rightarrow_R E'$. A rewrite relation defined on formulas can be extended to clauses as follows: $\mathcal{C} \Rightarrow_R \mathcal{C}'$ if $\mathcal{C}$ can be written as $\mathcal{D}, F$ and $\mathcal{C}'$ as $\mathcal{D}, F'$, for some clause $\mathcal{D}$ and formulas F and F' with $F \Rightarrow_R F'$.

Expressions that can not be rewritten are said to be in *normal form*. We write $E \Rightarrow^!_R E'$ to indicate that $E \Rightarrow^*_R E'$ and E' is in normal form. We say that R *terminates* if there is no infinite sequence $E_0 \Rightarrow_R E_1 \Rightarrow_R \cdots$ of rewrite steps. If R terminates, then every formula can be rewritten to a normal form (in zero or more steps).

If R and S are rewrite systems, we denote by R/S (R *modulo* S) the rewrite system consisting of all rules $E \Rightarrow E'$, such that $E \Leftrightarrow^*_S G \Rightarrow_R G' \Leftrightarrow^*_S E'$, for some expressions G and G'.

2.4. Refutational Theorem Proving

Theorem provers are procedures that can be used to check whether a given formula F (the “goal”) is a logical consequence of a set of formulas N (the “theory”). Refutational theorem provers deal with the equivalent problem of showing that the set $N \cup \{\neg F\}$ is inconsistent. The inconsistency of a set N may be established either by a semantic analysis or by providing a formal proof of $\bot$ from N, where proofs are traces of deductive inferences defined by a collection of inference rules.

For our purposes, an *inference rule* is an $n+1$-ary relation on general clauses.[3] The elements of such a relation are usually written as

$$\frac{\mathcal{C}_1 \quad \ldots \quad \mathcal{C}_n}{\mathcal{C}}$$

and called *inferences*. The clauses $\mathcal{C}_1, \ldots, \mathcal{C}_n$ are called the *premises*, and $\mathcal{C}$ the *conclusion*, of the inference. An *inference system* Γ is a collection of inference rules.

If I is an inference or a set of inferences we denote by $\mathsf{C}(\mathsf{I})$ its conclusion or the set of their conclusions. We also speak of an inference *from* a set of clauses N if all premises are elements of N, and denote by $\Gamma(N)$ the set of all inferences by Γ from N.

An inference is said to be *sound* if its conclusion is a logical consequence of its premises, $\mathcal{C}_1, \ldots, \mathcal{C}_n \models \mathcal{C}$. Soundness is often the minimal requirement expected in an inference system, but in refutational theorem proving it is sufficient that inferences preserve consistency. We call an inference system Γ *consistency-preserving* if for all sets of clauses N, the set $N \cup \mathsf{C}(\Gamma(N))$ is consistent whenever N is consistent. A sound inference system is consistency-preserving, but the converse is not true in general. We will only consider consistency-preserving inference systems.

[3] In refutational theorem proving inferences without premises, or axioms, are of little use, so that we always have $n \geq 1$.

For the inference systems that we study in this chapter, the order of the premises in an inference is relevant, and we view inferences as mechanisms whereby a distinguished premise (the *main premise*) is "reduced" to the conclusion in the context of the other premises (the *side premises*). Unless specified otherwise, the last premise of an inference is the main premise, and the other premises, if any, the side premises.

A *proof* of a clause $\mathcal{C}$ from a set of clauses N with respect to an inference system Γ is a sequence of clauses $\mathcal{C}_1, \ldots, \mathcal{C}_m$, such that $\mathcal{C} = \mathcal{C}_m$ and each clause $\mathcal{C}_i$ is either an element of N or else the conclusion of an inference by Γ from $N \cup \{\mathcal{C}_1, \ldots, \mathcal{C}_{i-1}\}$. The clauses in N are also called *assumptions*. We write $N \vdash_\Gamma \mathcal{C}$ if there exists a proof of $\mathcal{C}$ from N by Γ. If $\mathcal{C}$ is a contradiction we speak of a *refutation* of N.

An inference system Γ is said to be *refutationally complete* if there is a refutation by Γ from any unsatisfiable set of clauses N. A set of clauses N is called *saturated* with respect to Γ if the conclusion of any inference by Γ from N is an element of N. If an inference system Γ is refutationally complete and a set N is saturated with respect to Γ, then N is either satisfiable or contains a contradiction.

2.5. Orderings

Many theorem proving calculi employ orderings of one kind or another to obtain an approximate measure of the progress of a derivation towards a particular goal.

A (strict) *partial ordering* is a transitive and irreflexive binary relation; a *quasi-ordering* a reflexive and transitive binary relation. If $\succ$ is a strict ordering, its reflexive closure $\succeq$ is defined by: $x \succeq y$ if $x \succ y$ or $x = y$. A strict ordering is said to be *total* (on a subset S of the domain) if for any two distinct elements x and y (in S) we have either $x \succ y$ or $y \succ x$. The reflexive closure of a strict ordering is a quasi-ordering. On the other hand, if $\succsim$ is a quasi-ordering, then its *strict part* $\succ$ defined by: $x \succ y$ if $x \succsim y$ but not $y \succsim x$, is a strict ordering. We may also define an equivalence relation by: $x \sim y$ if $x \succsim y$ and $y \succsim x$. We say that an ordering $\succ'$ *extends* $\succ$ if the latter is a subset of the former, i.e., $x \succ' y$ whenever $x \succ y$.

For example, we may compare formulas by their size and define either a strict ordering: $F \succ G$ if G is shorter (as a string) than F; or a quasi-ordering: $F \succsim G$ if F is not shorter than G. If F and G are of the same length, we have $F \succsim G$ and $G \succsim F$, but neither $F \succ G$ nor $G \succ F$. This ordering extends the subformula relation. The example also shows that the reflexive closure of the strict part of a quasi-ordering may be different from the quasi-ordering.

A strict ordering $\succ$ is said to be *well-founded* if there is no infinite descending chain $x_1 \succ x_2 \succ \cdots$ of elements. If $\succ$ is a well-founded ordering on a set S, a property P is true for all elements of S whenever the implication "if $P(y)$, for each y in S such that $x \succ y$, then $P(x)$" holds for each x in S. This proof principle is called *Noetherian* or *well-founded induction.* A quasi-ordering is well-founded if its strict part is well-founded.

We say that an ordering $\succ$ has the *subterm property* if $E[E'] \succ E'$, for all expressions E and proper subexpressions E' of E. A *rewrite ordering* is an ordering that is also a rewrite relation; a *reduction ordering*, a well-founded rewrite ordering;

and a *simplification ordering*, a reduction ordering with the subterm property. Note that a rewrite system R terminates if, and only if, there exists a reduction ordering $\succ$ such that $E\sigma \succ E'\sigma$, for each rule $E \Rightarrow E'$ in R and each substitution σ. Reduction orderings must be compatible with the subterm ordering in that $E' \not\succ E[E']$ for all expressions E and subexpressions E' of E. Thus, if E' and $E[E']$ are comparable with respect to a reduction ordering $\succ$, then $E[E'] \succeq E'$.

In theorem proving applications orderings are often defined with respect to the tree structure of terms and formulas. Let $\succ$ be an ordering, called a *precedence*, on the given set of (function, predicate and logical) symbols. The corresponding *lexicographic path ordering* $\succ_{lpo}$ is defined by:

$s = f(s_1, \ldots, s_m) \succ_{lpo} g(t_1, \ldots, t_n) = t$ if and only if
(i) $f \succ g$ and $s \succ_{lpo} t_i$, for all i with $1 \leq i \leq n$; or
(ii) $f = g$ and, for some j, we have $(s_1, \ldots, s_{j-1}) = (t_1, \ldots, t_{j-1})$, $s_j \succ_{lpo} t_j$, and $s \succ_{lpo} t_k$, for all k with $j < k \leq n$; or
(iii) $s_j \succsim_{lpo} t$, for some j with $1 \leq j \leq m$.

If the precedence is well-founded, the lexicographic path ordering is a simplification ordering. It is total on closed expressions whenever the precedence is total. For a survey on termination orderings see [Dershowitz 1987].

Any ordering on a set N can be extended to an ordering $\succ_{mul}$ on finite multisets over N as follows: $\Sigma_1 \succ_{mul} \Sigma_2$ if (i) $\Sigma_1 \neq \Sigma_2$ and (ii) whenever $\Sigma_2(x) > \Sigma_1(x)$ then $\Sigma_1(y) > \Sigma_2(y)$, for some y such that $y \succ x$. (Here $>$ denotes the standard "greater-than" relation on the natural numbers.) Given a multiset, any smaller multiset is obtained by (repeatedly) replacing an element by zero or more occurrences of smaller elements. If an ordering $\succ$ is total (resp., well-founded), so is its multiset extension. For example, if we order formulas by their size, then $P(f(a)) \succ Q(a)$ and $\{P(f(a))\} \succ_{mul} \{P(a), Q(a)\}$. For simplicity we often use the same symbol to denote both an ordering and its multiset extension.

If Σ is a multiset over M and x an element in M, x is said to be *[strictly] maximal* with respect to Σ if there is no element y in Σ such that $y \succ x$ $[y \succeq x]$. Since clauses are multisets, we may thus speak of the maximal formula in a general clauses, or the maximal literal in a standard clause. In addition, we say that A is a *maximal atom* of a clause $\mathcal{C}$ if A occurs in $\mathcal{C}$ and there is no atom B in $\mathcal{C}$ with $B \succ A$.

A reduction ordering $\succ$ that is total on closed formulas is called *admissible* if (i) $A \succ \top$ and $A \succ \bot$, for all atoms A; and (ii) $F \succ G$, whenever for all atoms B in G there exists an atom A in F such that $A \succ B$. An ordering $\succ$ on ground clauses is *admissible* if it is the multiset extension of an admissible ordering on formulas. If an ordering on clauses is admissible, then it is well-founded and total on ground clauses.

A lexicographic path ordering over a total precedence is admissible if predicate symbols have higher precedence than logical symbols and the constants $\top$ and $\bot$ are smaller than the other logical symbols. If one regards ground atoms as constants, and uses a well-founded ordering $\succ$ on atoms as a precedence, extended by $A \succ \equiv \succ \supset \succ \neg \succ \vee \succ \wedge \succ \top \succ \bot$, then the corresponding lexicographic path ordering is admissible. In other words, a well-founded ordering on ground atoms can always be extended to an admissible ordering on clauses.

3. Standard Resolution

Saturation-based theorem proving refers to a process in which two levels of data structures are manipulated. At the *level of deduction* new formulas are derived from given ones by applying specified inference rules, with the ultimate goal of obtaining a contradiction. In addition, the current set of formulas is analyzed to identify the most promising inference rules to be applied next and to eliminate redundancies. We will formalize this second level in terms of *theorem proving derivations*. The set of retained formulas represents the logical information that is explicitly available to a prover at a specific point in time, and provides the information used to determine further steps in the theorem proving process. Practical experience with theorem provers has shown that powerful and efficient "global" techniques for simplification of the current collection of formulas, and especially the elimination of redundancies therein, are far more important to an acceptable performance than any "local" refinements of the deductive inferences at the formula level. But inference computation and redundancy elimination interact in subtle ways and one has to be careful that their integration is not counterproductive and that refutational completeness is preserved.

In this chapter, and in more detail in Chapter 5 we will investigate resolution-based theorem proving methods for standard variable-free clauses. A standard clause is a multiset of literals. We say that an atom A occurs *positively* in a clause $\mathcal{C}$, if A is one of the literals of $\mathcal{C}$, and occurs *negatively* if $\neg A$ is a literal in $\mathcal{C}$.

Numerous versions of resolution for standard clauses have been proposed in the literature, most of which will be discussed in later chapters. The following variant, for ground clauses, combines factoring and resolution into a single inference rule.

Binary resolution with factoring

$$\frac{C \vee A \vee \cdots \vee A \quad \neg A \vee D}{C \vee D}$$

We speak of a *resolution on* A and call the conclusion of the inference a *resolvent* of the two premises. The main premise of the inference is $\neg A \vee D$, while $C \vee A \vee \ldots \vee A$ is the side premise. By B we denote the set of all inferences by binary resolution with factoring. The calculus B is used in Figure 1 to derive a contradiction from five given input clauses.

Resolution is a sound inference. Suppose I is an interpretation in which both premises are valid. The atomic formula A is either true or false in I. If A is true in I then D must be true in I, for otherwise the main premise would be false in I. Similarly, if $\neg A$ is true in I then C must be true in I. In either case, the resolvent will be true in I.

Binary resolution with factoring is also refutationally complete, which we prove by showing that any inconsistent set of clauses that is closed under B contains a contradiction. In its contrapositive form this statement asserts that any set of clauses that is closed, but contains no contradiction, has a model. We specify a Herbrand model by induction on a suitably chosen clause ordering.

(1)	$\neg A \vee B$	[input]
(2)	$\neg B \vee C$	[input]
(3)	$A \vee \neg C$	[input]
(4)	$A \vee B \vee C$	[input]
(5)	$\neg A \vee \neg B \vee \neg C$	[input]
(6)	$A \vee B \vee A$	[resolving on C in (4) and (3)]
(7)	$B \vee B$	[resolving on A in (6) and (1)]
(8)	C	[resolving on B in (7) and (2)]
(9)	A	[resolving on C in (8) and (3)]
(10)	$\neg B \vee \neg C$	[resolving on A in (9) and (5)]
(11)	$\neg C$	[resolving on B in (7) and (10)]
(12)	$\perp$	[resolving on C in (8) and (11)]

Figure 1: A sample refutation

Let $\succ$ be a total admissible ordering on clauses. Given a set of clauses N, we use induction with respect to $\succ$ to define for each clause C (not necessarily in N) a Herbrand interpretation I_C and a set ε_C as follows.

3.1. Definition. Take I_C to be the set $\bigcup_{C \succ D} \varepsilon_D$. Furthermore, if C is a clause that
(i) is contained in N;
(ii) is of the form $C' \vee A$, where A is the maximal literal in C; and
(iii) is false in I_C;
then $\varepsilon_C = \{A\}$; otherwise, ε_C is the empty set.

We also say that C *produces* A, and call C a *productive clause*, if $\varepsilon_C = \{A\}$. I_C is called the *partial interpretation below* C. By the *partial interpretation at* C we mean the (possibly extended) set $I^C = I_C \cup \varepsilon_C$. Finally, by the *candidate model* for N, denoted by $I_N^{\succ}$ or simply I_N, we mean the Herbrand interpretation $\bigcup_{C \in N} \varepsilon_C$.

The partial interpretation I_C is intended to be a model of the set N_C of those clauses in N that are smaller than C (with respect to the given clause ordering); whereas ε_C is meant to be a minimal extension of I_C that makes C true. Only clauses in N in which the maximal atom A is positive, can possibly be productive. We say that such a clause is *reductive* for A. Reductive clauses may be viewed as implications $\neg C \supset A$. The recursive evaluation of the condition $\neg C$ (via "negation as failure") will eventually terminate. If the condition $\neg C$ evaluates to true and A is not yet contained in the corresponding partial interpretation, then extending the interpretation by A will make the reductive clause true, but does not affect the truth values of the (smaller) clauses which were used to evaluate its condition $\neg C$.

The following example shows that the construction need not always yield a model of (non-productive clauses in) N.

3.2. EXAMPLE. Take an ordering $B_2 \succ A_2 \succ B_1 \succ A_1 \succ B_0 \succ A_0$ on atoms. The following table describes the various partial interpretations for clauses, listed in ascending order.

clause C	interpretation I_C	ε_C	remarks
$A_0 \vee B_0$	$\emptyset$	$\{B_0\}$	B_0 is maximal
$B_0 \vee A_1$	$\{B_0\}$	$\emptyset$	true in I_C
$\neg B_0 \vee A_1$	$\{B_0\}$	$\{A_1\}$	A_1 is maximal
$\neg B_0 \vee A_2 \vee B_1$	$\{B_0, A_1\}$	$\{A_2\}$	A_2 is maximal
$\neg B_0 \vee \neg A_2 \vee B_1$	$\{B_0, A_1, A_2\}$	$\emptyset$	B_1 not maximal
$\neg B_1 \vee B_2$	$\{B_0, A_1, A_2\}$	$\emptyset$	true in I_C

The next to last clause is false in the final interpretation $\{B_0, A_1, A_2\}$.

The following lemmas clarify key connections between candidate models and clauses.

3.3. LEMMA. *If C is productive, then C is true in I_N.*

PROOF. By the construction, if C produces an atom A then C is true in $I_C \cup \{A\}$. Since $I_C \cup \{A\} \subseteq I_N$, the clause C is also true in I_N. □

3.4. LEMMA. *Let C and D be clauses such that $D \succeq C$. If C is true in I_D or I^D then C is also true in I_N and in all interpretations $I_{D'}$ and $I^{D'}$, where $D' \succeq D$.*

PROOF. First, observe that $I_D \subseteq I^D \subseteq I_{D'} \subseteq I^{D'} \subseteq I_N$, whenever $D' \succeq D$. If C contains a positive literal A that is true in I_D or I^D, then A is also true in I_N and in all interpretations $I_{D'}$ and $I^{D'}$, so that the assertion follows immediately. Otherwise, C must contain a negative literal $\neg A$, such that A is false in I_D. Suppose there is a clause D'' that is reductive for A. Then A is the maximal literal in D'', but since admissible orderings are reduction orderings and total they satisfy the subterm property, so that $\neg A \succ A$, and hence $D \succeq C \succeq \neg A \succ D''$. This contradicts the fact that A is false in I_D. We conclude that A is false, and $\neg A$ and C are true, in I_N and in all interpretations $I_{D'}$ and $I^{D'}$. □

We often use this lemma in its contrapositive form to infer that if a clause C is false some interpretation I_N or $I_{D'}$ or $I^{D'}$, and D is a clause with $D' \succeq D \succeq C$, then C is also false in I_D and in I^D.

3.5. LEMMA. *If C is a clause in N, the maximal literal of which is positive, then C is true in I_N.*

Proof. If the maximal literal of a clause C in N is positive, then the clause is either productive or else is true in I_C. By the above lemmas, the clause is true in I_N. □

3.6. Lemma. *Let D and D' be clauses such that $D \succeq D'$ and either D' is a clause in N or else the maximal atom in D is strictly greater than the maximal atom in D'. If D' is false in I^D, then it is also false in I_N and in all interpretations I_C and I^C, where $C \succ D$.*

Proof. Let D and D' be clauses as specified. Suppose D' is false in I^D, but true in I_N or in some interpretation I_C or I^C, where $C \succ D$. This is only possible if D' contains a positive occurrence of an atom A that is produced by some clause $C' \succ D$. But in that case A must be the maximal atom in D', so that D' can not be in N, for otherwise it would not be false in I^D by the above lemmas. Therefore A must be strictly smaller than the maximal atom of D, which contradicts the assumption that C' is a productive clause for A with $C' \succ D$. □

3.7. Lemma. *If D is a clause in N and another clause C is true in all interpretations $I^{D'}$, where $D \succ D'$, then D' is also true in I_D.*

Proof. Let C and D be clauses as specified. If some atom that occurs positively in C is produced by a clause strictly smaller than D, then C is obviously true in I_D. Suppose, on the other hand, that none of the positive atoms in C, but all of the atoms with negative occurrences, are produced by clauses strictly smaller than D. Let A be the maximal atom that occurs negatively in C, and C' be the clause that produces A. Then C is already false in $I^{C'}$, contradicting our assumption. We may conclude that whenever none of the positive atoms in C is true in I_D, then some atom with a negative occurrence is false in I_D, making C true in I_D. □

We have seen in the above example that non-productive clauses may be false in I_N. A clause that is false in an interpretation I is also called a *counterexample* for I. If a set N contains a counterexample for I_N, then it must also contain a minimal counterexample for I_N (with respect to the admissible clause ordering $\succ$). The following key result indicates that a minimal counterexample is either a contradiction or else can be reduced to an even smaller counterexample by resolution.

3.8. Theorem. *Let N be a set of clauses not containing the empty clause and C be a minimal counterexample in N for I_N. Then there exists an inference by binary resolution with factoring from C such that*
(i) its conclusion is a counterexample for I_N and is smaller than C; and
(ii) C is its main premise and the side premise is a productive clause.

Proof. For simplicity let us denote I_N by I. By Lemma 3.3, productive clauses are true in I. Therefore, the minimal counterexample C must be a non-productive clause. Since N does not contain the empty clause, C must contain at least one literal. Let A be its maximal atom, which must occur negatively for C to be a counterexample for I. Thus C can be written as $\neg A \vee C'$, where A is true in I. Let $D = D' \vee A \vee \ldots \vee A$ be the clause that produces A, with a subclause D' that does not contain A. The clause D is reductive for A, so that all atoms in D' are strictly smaller than A. Moreover, D and, hence, D' are false in I_D. Therefore, resolution with the productive clause D as side premise and C as main premise yields a resolvent $D' \vee C'$ that is strictly smaller than C. By Lemma 3.6, D' is false in I, and consequently $D' \vee C'$ is false in I. In short, resolution yields a smaller counterexample than C. □

3.9. THEOREM. *If N is saturated with respect to* B *and does not contain the empty clause then I_N is a model of N.*

PROOF. Theorem 3.8 indicates that the only counterexample that can not be reduced to an even smaller counterexample by resolution is the empty clause. For saturated sets N there are thus only two possibilities: the set either contains no counterexample (so that I_N is a model of N), or else contains the empty clause. □

The theorem also implies that B is refutationally complete, for if the empty clause can not be derived, then N has a model.

3.10. COROLLARY. *The inference system* B *is refutationally complete.*

3.11. EXAMPLE. Continuing Example 3.2 we observe that $\neg B_0 \vee \neg A_2 \vee B_1$ is a smallest counterexample, with A_2 as maximal atom. The atom A_2 is produced by $\neg B_0 \vee A_2 \vee B_1$. Resolving the two clauses yields a smaller counterexample $\neg B_0 \vee B_1 \vee \neg B_0 \vee B_1$ to the initial candidate model. A modified model construction with this additional resolvent yields

clause C	interpretation I_C	ε_C	remarks
$A_0 \vee B_0$	$\emptyset$	$\{B_0\}$	B_0 is maximal
$B_0 \vee A_1$	$\{B_0\}$	$\emptyset$	true in I_C
$\neg B_0 \vee A_1$	$\{B_0\}$	$\{A_1\}$	A_1 is maximal
$\neg B_0 \vee B_1 \vee \neg B_0 \vee B_1$	$\{B_0, A_1\}$	$\{B_1\}$	B_1 is maximal
$\neg B_0 \vee A_2 \vee B_1$	$\{B_0, A_1, B_1\}$	$\emptyset$	true in I_C
$\neg B_0 \vee \neg A_2 \vee B_1$	$\{B_0, A_1, B_1\}$	$\emptyset$	
$\neg B_1 \vee B_2$	$\{B_0, A_1, B_1\}$	$\{B_2\}$	

The resulting interpretation $I = \{B_0, A_1, B_1, B_2\}$ is a model of all clauses.

Another consequence of Theorem 3.9 is the compactness of closed clausal logic.

3.12. THEOREM. *A set N of ground clauses is unsatisfiable if and only if some finite subset of N is unsatisfiable.*

Ordered resolution with selection

$$\frac{C_1 \vee A_1 \vee \cdots \vee A_1 \quad \ldots \quad C_n \vee A_n \vee \cdots \vee A_n \quad \neg A_1 \vee \cdots \vee \neg A_n \vee D}{C_1 \vee \cdots \vee C_n \vee D}$$

where
(i) either the subclause $\neg A_1 \vee \cdots \vee \neg A_n$ is selected by S in D, or else $S(D)$ is empty, $n = 1$, and A_1 is maximal with respect to D,
(ii) each atom A_i is strictly maximal with respect to C_i, and
(iii) no clause $C_i \vee A_i \vee \ldots \vee A_i$ contains a selected atom.

Figure 2: *(Standard) Ordered Resolution* $\mathsf{O}_S^{\succ}$

Resolution, as a sound and complete inference rule, provides a suitable deductive basis for refutational theorem proving. For practical purposes the above inference rule is too prolific, though, in that too many clauses can be deduced (the "search space" is too large). We next discuss useful restrictions on resolution that do not impair its completeness.

An inspection of the proof of Theorem 3.8 reveals that we have actually established a stronger result than is stated in the theorem. For instance, resolution is only required on the maximal atom in the side premise, so that corresponding ordering restrictions may be imposed on inferences. Furthermore, it is sufficient to resolve on any of the negative literals in a don't-care non-deterministic way. We propose selection functions as a corresponding control mechanism. Finally, we will package several inferences into one larger inference step by simultaneously resolving on more than one atom. This has the possible advantage that intermediate results need not be retained.

By a *selection function* we mean a mapping S that assigns to each clause C a (possibly empty) multiset $S(C)$ of negative literals in C. In other words, the function S selects (a possibly empty) negative subclause of C. We say that an atom A, or a literal $\neg A$, is *selected by* C if $\neg A$ occurs in $S(C)$. (There are no selected atoms or literals if $S(C)$ is empty.)

Let $\succ$ be an admissible clause ordering and S be a selection function. The inference system $\mathsf{O}_S^{\succ}$ of ordered resolution with selection is shown in Figure 2. (The subscript and/or superscript in $\mathsf{O}_S^{\succ}$ will be omitted if the relevant information is either clear from the context or intentionally left unspecified. Specific settings of the parameters will be discussed in Sections 6 and 7.) In ordered inferences one resolves either all selected atoms at once or, in case there are no selected atoms, the maximal atom of the main premise. Furthermore, the side premises must contain no selected atoms at all.

A key argument in the proof of Theorem 3.8 is that resolution inferences can be used to reduce certain counterexamples. Ordered resolution is designed so that each resolvent is smaller than the corresponding main premise.

3.13. LEMMA. *If $\succ$ is an admissible ordering and S any selection function, then the conclusion of any inference in $\mathsf{O}_S^{\succ}$ is smaller (with respect to $\succ$) than the main premise.*

Theorem 3.8 can easily be extended to ordered resolution with selection by applying similar ideas to a model construction that is slightly modified from Definition 3.1. Given $\succ$, a total admissible ordering on clauses, and a set of clauses N, we again use induction with respect to $\succ$ to define for each clause C (not necessarily in N) a Herbrand interpretation I_C and a set ε_C as follows.

3.14. DEFINITION. Take I_C to be the set $\bigcup_{C \succ D} \varepsilon_D$. Furthermore, if C is a clause that
(i) is contained in N;
(ii) is of the form $C' \vee A$, where A is the maximal literal in C;
(iii) is false in I_C; and
(iv) nothing is selected in C;
then $\varepsilon_C = \{A\}$; otherwise, ε_C is the empty set.

Again we say that C *produces* A, and call C a *productive clause*, if $\varepsilon_C = \{A\}$. By the defn[candidate model]*candidate model* for N, denoted by $I_N^{\succ}$ or simply I_N, we now mean the Herbrand interpretation $\bigcup_{C \in N} \varepsilon_C$ as just defined. As the side-premises of counterexample-reducing inferences are productive clauses, they should not have any selected literals. A related modification of the proof of Theorem 3.8 gives us this refined result:

3.15. THEOREM. *Let N be a set of clauses not containing the empty clause. Let C be the minimal counterexample in N for I_N. Then there exists an inference in O from C such that*
(i) its conclusion is a counterexample for I_N and is smaller than C; and
(ii) C is its main premise and the side premises are productive clauses.

This reduction property for counterexamples by ordered resolution immediately implies the refutational completeness of the inference system.

3.16. THEOREM. *If N is saturated with respect to O and does not contain the empty clause then I_N is a model of N. The inference system O is therefore refutationally complete.*

This concludes our introduction to saturation-based theorem proving. More sophisticated techniques will be discussed in later chapters. But first we need to outline a rigorous framework for the description of theorem proving strategies.

4. A Framework for Saturation-Based Theorem Proving

All theorem provers employ some deductive inference mechanisms. In the case of refutational provers the goal is to derive a contradiction from any inconsistent set of input formulas, and the derivation process typically amounts to a saturation of the input set. A straightforward, naive saturation in which one exhaustively applies inferences to previously derived clauses will be hopelessly inefficient in all but the most trivial cases. In a clausal prover each derived clause represents a partial proof or proof attempt, and increases the number of possibilities for constructing a complete proof (or refutation). A (partial) proof (attempt) that is subsumed by another is redundant and should be deleted to avoid useless computations. In most refutational provers, the deductive core accounts for a rather small part of the system, while most of the complexity of the prover derives from the implementation of powerful, yet efficient redundancy elimination and simplification techniques. The degree of sophistication of the latter tools usually distinguishes experimental prototypes from practically useful tools. Unfortunately, comparatively little effort has been devoted to a formal analysis of redundancy and other fundamental concepts of theorem proving strategies, while more emphasis has been placed on investigating the refutational completeness of a variety of modifications of inference rules, such as resolution.

We will next describe a comprehensive framework for modeling the key aspects of theorem proving, such as deduction, deletion, and simplification. In this formalism we will be able to describe a wide range of different theorem proving strategies and to argue about their refutational completeness. The concepts and results in this chapter do not depend on details pertaining to specific syntax or representation of formulas, but apply to *general* clauses containing arbitrary quantifier-free subformulas.

4.1. Theorem Proving Processes

We first develop the minimal prerequisites for a theory of refutational theorem proving with deduction and deletion. We assume that deduction is based on a clausal inference system Γ. The formalization of deletion is more subtle and technically involved in that formulas may only be deleted if one can be certain that this will not prevent the successful completion of the proof search. We will formulate deletion strategies in terms of redundancy criteria for formulas and inferences. Redundancy refers to the states of the theorem proving process, as represented by the collection of clauses that have been derived and retained.

A redundancy criterion is specified by two mappings $\mathcal{R}_\mathcal{F}$ and $\mathcal{R}_\mathcal{I}$, which associate with each set N of clauses a set of clauses and a set of inferences, respectively, that are deemed to be redundant in the context N.

For example, $\mathcal{R}_\mathcal{F}(N)$ will usually contain all tautologies (in the given language), whereas $\mathcal{R}_\mathcal{I}(N)$ may contain all inferences the conclusion of which is already an element of N.

4.1. Definition. Let Γ be an inference system. A pair $\mathcal{R} = (\mathcal{R}_\mathcal{F}, \mathcal{R}_\mathcal{I})$ of mappings from sets of clauses to sets clauses and inferences (by Γ), respectively, is called a *redundancy criterion* if, for all sets of clauses N and N'

Deduction
$N \rhd N,\ M$ if $M \subseteq \mathsf{C}(\Gamma(N))$

Deletion
$N,\ M \rhd N$ if $M \subseteq \mathcal{R}(N)$

Figure 3: *Theorem Proving Derivations* $\rhd$

(R1) if $N \subseteq N'$ then $\mathcal{R}_{\mathcal{F}}(N) \subseteq \mathcal{R}_{\mathcal{F}}(N')$ and $\mathcal{R}_{\mathcal{I}}(N) \subseteq \mathcal{R}_{\mathcal{I}}(N')$;
(R2) if $N' \subseteq \mathcal{R}_{\mathcal{F}}(N)$ then $\mathcal{R}_{\mathcal{F}}(N) \subseteq \mathcal{R}_{\mathcal{F}}(N \setminus N')$ and $\mathcal{R}_{\mathcal{I}}(N) \subseteq \mathcal{R}_{\mathcal{I}}(N \setminus N')$; and
(R3) if N is inconsistent, then $N \setminus \mathcal{R}_{\mathcal{F}}(N)$ is also inconsistent.
The criterion is called *effective* (for Γ) if, in addition,
(R4) an inference γ in Γ is in $\mathcal{R}_{\mathcal{I}}(N)$ whenever its conclusion is in $N \cup \mathcal{R}_{\mathcal{F}}(N)$.

The first condition expresses monotonicity of redundancy under the subset relation and, in particular, under the deduction of new clauses. The second condition requires that redundancy be independent of clauses that are redundant in the given context. The third condition states that the removal of redundant clauses preserves inconsistency. Finally, the fourth condition implies that adding its conclusion renders an inference redundant, so that redundancy of inferences can always be achieved by systematic computation of inferences. Inferences in $\mathcal{R}_{\mathcal{I}}(N)$ and clauses in $\mathcal{R}_{\mathcal{F}}(N)$, respectively, are said to be *redundant* (with respect to $\mathcal{R}$ in context N). We emphasize that $\mathcal{R}_{\mathcal{F}}(N)$ need not be a subset of N and that $\mathcal{R}_{\mathcal{I}}(N)$ may contain inferences whose premises are not in N.

A *trivial*, but effective, redundancy criterion for any inference system Γ is given by the mappings $\mathcal{R}_{\mathcal{F}}(N) = \emptyset$ and $\mathcal{R}_{\mathcal{I}}(N) = \{\gamma \in \Gamma \mid \mathsf{C}(\gamma) \in N\}$. That is, no clause is redundant in any context, while an inference is taken to be redundant only if its conclusion is already present. More interesting, non-trivial redundancy criteria for resolution-based inference systems will be described later.

At an abstract level a saturation-based theorem prover can be described by a binary relation $\rhd$ on sets of clauses, called a *transition* or *derivation relation*. We specifically consider derivation relations where each step $N \rhd N'$ consists of either adding logical consequences (by applying inferences from Γ) or deleting redundant clauses (according to a criterion $\mathcal{R}$ for Γ), cf. Figure 3 (where we use multiset notation for clauses and write, for instance, N, M instead of $N \cup M$). A (finite or countably infinite) sequence $N_0 \rhd N_1 \rhd N_2 \rhd \cdots$ is called a *(theorem proving) derivation* (based on Γ and $\mathcal{R}$). The set $N_\infty = \bigcup_i \bigcap_{j \geq i} N_j$ of all *persisting clauses* is called the *limit* of the derivation. By a *theorem prover* we mean a procedure that accepts as input a set of clauses N, and produces a derivation $N = N_0 \rhd N_1 \rhd N_2 \rhd \cdots$ from N based on a given inference system Γ and redundancy criterion $\mathcal{R}$. The sets N_i represent the successive states in the theorem proving process; the set N_∞ its result (which in the case of an infinite derivation is only obtained in the limit).

We consider deductive rules that are consistency-preserving, and condition (R3) ensures that the deletion of redundant formulas preserves the inconsistency of a set of clauses. It can therefore easily be seen that in a theorem proving derivation either all clause sets N_i are consistent, or all sets are inconsistent.

4.2. Lemma. *Let $N_0 \rhd N_1 \rhd N_2 \rhd \cdots$ be a derivation based on a Γ and $\mathcal{R}$. Then $\mathcal{R}_\mathcal{F}(\bigcup_j N_j) \subseteq \mathcal{R}_\mathcal{F}(N_\infty)$ and $\mathcal{R}_\mathcal{I}(\bigcup_j N_j) \subseteq \mathcal{R}_\mathcal{I}(N_\infty)$. Moreover, the limit set N_∞ is satisfiable if and only if the initial set N_0 is satisfiable.*

Proof. First note that by the definition of N_∞, a clause that is in $(\bigcup_j N_j)$ but not in N_∞, must be in some set $\mathcal{R}_\mathcal{F}(N_i)$. Therefore we have $(\bigcup_j N_j) \setminus N_\infty \subseteq \bigcup_j \mathcal{R}_\mathcal{F}(N_j)$. Moreover, by condition (R1), $\bigcup_j \mathcal{R}_\mathcal{F}(N_j) \subseteq \mathcal{R}_\mathcal{F}(\bigcup_j N_j)$. As a consequence, we have $(\bigcup_j N_j) \setminus \mathcal{R}_\mathcal{F}(\bigcup_j N_j) \subseteq N_\infty$. Applying condition (R1) again, we may infer that $\mathcal{R}((\bigcup_j N_j) \setminus \mathcal{R}_\mathcal{F}(\bigcup_j N_j)) \subseteq \mathcal{R}(N_\infty)$ (where $\mathcal{R}$ may be either $\mathcal{R}_\mathcal{F}$ or $\mathcal{R}_\mathcal{I}$). Using condition (R2) we obtain $\mathcal{R}(\bigcup_j N_j) \subseteq \mathcal{R}(N_\infty)$.

For the second part, note that by the soundness of the inference system Γ and the compactness of clausal logic, the set N_0 is satisfiable if and only if $\bigcup_j N_j$ is satisfiable. Thus, if N_0 is satisfiable, then N_∞, as a subset of $\bigcup_j N_j$, is also satisfiable. On the other hand, if N_0 is unsatisfiable, then by condition (R3) the set $(\bigcup_j N_j) \setminus \mathcal{R}_\mathcal{F}(\bigcup_j N_j)$ is unsatisfiable as well, and N_∞ is a superset. □

A refutationally complete theorem prover derives a contradiction from any inconsistent initial set N_0. Evidently "sufficiently many" inferences must be computed to ensure refutational completeness; a more precise characterization is based on two important concepts, saturation and fairness.

We say that N is *saturated up to redundancy* (with respect to Γ and $\mathcal{R}$) if all inferences in Γ with non-redundant premises from N are redundant in N, i.e., $\Gamma(N \setminus \mathcal{R}_\mathcal{F}(N)) \subseteq \mathcal{R}_\mathcal{I}(N)$. For example, a set N is saturated with respect to the trivial redundancy criterion $\mathcal{R}$ if and only if $\mathsf{C}(\Gamma(N)) \subseteq N$. In general, saturation up to redundancy can be achieved by a "fair" computation of inferences from persisting, non-redundant clauses.

A derivation $N_0 \rhd N_1 \rhd N_2 \rhd \cdots$ based on an inference system Γ and an *effective* redundancy criterion $\mathcal{R}$ is called *fair* if the conclusion of every non-redundant inference in Γ from non-redundant formulas in N_∞ is either an element of, or redundant in, $\bigcup_j N_j$; that is, if $N' = N_\infty \setminus \mathcal{R}_\mathcal{F}(N_\infty)$ then

$$\mathsf{C}(\Gamma(N') \setminus \mathcal{R}_\mathcal{I}(N')) \subseteq \bigcup_j N_j \cup \mathcal{R}_\mathcal{F}(\bigcup_j N_j).$$

Intuitively fairness means that no inference in Γ from non-redundant persisting formulas be delayed indefinitely. A fair derivation can be constructed by exhaustively applying inferences to persisting formulas.

4.3. Theorem. *If a derivation, based on an inference system Γ and an effective redundancy criterion $\mathcal{R}$, is fair then its limit is saturated up to redundancy with respect to Γ and $\mathcal{R}$.*

PROOF. Let γ be an inference from non-redundant clauses in N_∞ and $\mathcal{C}$ its conclusion. If $\mathcal{C}$ is in $\bigcup_j N_j$ then, by the condition (R4), γ is redundant in $\bigcup_j N_j$ and, by Lemma 4.2, also redundant in N_∞. On the other hand, if the clause $\mathcal{C}$ is redundant in $\bigcup_j N_j$ and, hence, in N_∞, then by condition (R4) the inference γ is redundant in N_∞. We conclude that N_∞ is saturated up to redundancy. □

Fairness provides an effective way of saturation for effective redundancy criteria.

Propositional inference systems such as $O_S^\succ$ can only be approximately lifted to non-ground clauses. That is, the corresponding non-ground versions of the inference rules usually have more ground instances than needed. Fortunately, the concepts of redundancy and theorem proving derivations are insensitive to such extensions of the inference system. If Γ' is an inference system extending Γ, that is, $\Gamma \subseteq \Gamma'$, then any redundancy criterion $\mathcal{R}$ for Γ can be extended to a redundancy criterion $\mathcal{R}'$ for Γ' by defining $\mathcal{R}'_{\mathcal{F}}(N) = \mathcal{R}_{\mathcal{F}}(N)$ and $\mathcal{R}'_{\mathcal{I}}(N) = \mathcal{R}_{\mathcal{I}}(N) \cup (\Gamma' \setminus \Gamma)$, for all sets of clauses N. With this definition, the additional inferences in $\Gamma' \setminus \Gamma$ become redundant in any context. This *standard extension* of a redundancy criterion is useful if the inferences in $\Gamma' \setminus \Gamma$ are optional for a theorem prover. If $\mathcal{R}$ is effective, so is its standard extension $\mathcal{R}'$. We say that a *derivation is based on an extension* of Γ and $\mathcal{R}$ whenever there is an inference system $\Gamma' \supseteq \Gamma$ such that the derivation is based on Γ' and the standard extension of $\mathcal{R}'$. In that case, if the derivation is fair with respect to inferences in Γ the derivation is also fair with respect to inferences in Γ', and vice versa. The limit of any such derivation is, therefore, saturated with respect to Γ and $\mathcal{R}$.

4.2. Counterexample-Reducing Inference Systems

Candidate models and reduction of counterexamples are key concepts in establishing the refutational completeness of refutational theorem proving systems. In making these concepts more explicit, we will also arrive at a useful notion of redundancy for resolution. Throughout this section we assume that all expressions are ground and that $\succ$ is an admissible clause ordering. Clauses with variables will be discussed later.

4.2.1. Candidate Models and Counterexamples

Let I be a mapping, called a *model functor*, that assigns to each set N of ground clauses not containing a contradiction an interpretation I_N, called a *candidate model*. If I_N is a model of N, then N is evidently satisfiable. If, on the other hand, some clause $\mathcal{C}$ in N is false in I_N (a *counterexample* for I_N), then N must contain a *minimal* such counterexample with respect to $\succ$. We say that an inference system Γ has the *reduction property for counterexamples* (with respect to I and $\succ$) if, for all sets N of clauses and minimal counterexamples $\mathcal{C}$ for I_N in N, there exists an inference in Γ from N with main premise $\mathcal{C}$, side premises that are true in N, and a conclusion $\mathcal{D}$ that is a smaller counterexample for I_N than $\mathcal{C}$, i.e., $\mathcal{C} \succ \mathcal{D}$. Inference systems with this property are refutationally complete (with respect to the trivial redundancy criterion).

4.4. Theorem. *If* Γ *has the reduction property for counterexamples and* N *is saturated with respect to* Γ*, that is,* $\mathsf{C}(\Gamma(N)) \subseteq N$*, then* N *is either satisfiable or else contains the empty clause.*

Proof. Suppose Γ has the reduction property for counterexamples with respect to some model functor I, and N is a set of clauses that does not contain the empty clause. If N contains a counterexample for I_N, it also contains a minimal counterexample $\mathcal{C}$. By the reduction property, $\mathcal{C}$ can be reduced to an even smaller counterexample $\mathcal{D}$, which contradicts the minimality of $\mathcal{C}$. Thus N can not contain a counterexample for I_N, which implies that I_N is a model of N. □

4.2.2. The Standard Redundancy Criterion

Inference systems with the reduction property for counterexamples admit a non-trivial redundancy criterion, called the *standard redundancy criterion*, that is based on the given clause ordering, but largely independent of the inference system. The intention in defining the criterion is to identify those clauses and inferences which cannot involve any minimal counterexample.

A clause $\mathcal{C}$ is called *redundant* with respect to a set N of clauses if there exist clauses $\mathcal{C}_1, \ldots, \mathcal{C}_k$ in N such that $\mathcal{C}_1, \ldots, \mathcal{C}_k \models \mathcal{C}$ and $\mathcal{C} \succ \mathcal{C}_i$, for all i with $1 \leq i \leq k$. Note that $\mathcal{C}$ need not be an element of N. For example, tautological clauses $C \vee A \vee \neg A$ are redundant in any context N. We define a mapping $\mathcal{R}^{\succ}_{\mathcal{F}}$ by taking $\mathcal{R}^{\succ}_{\mathcal{F}}(N)$ to be the set of all redundant clauses with respect to N. By $N_{\mathcal{C}}$ we denote the set of clauses in N that are smaller than $\mathcal{C}$. A redundant clause $\mathcal{C}$ logically follows from $N_{\mathcal{C}}$, and therefore can not be a minimal counterexample in N for any interpretation.

If an inference reduces a minimal counterexample (i.e., the main premise), then its conclusion, but none of the side premises, is a counterexample, which suggests the following definition. An inference with main premise $\mathcal{C}$, side premises $\mathcal{C}_1, \ldots, \mathcal{C}_n$, and conclusion $\mathcal{D}$ is called *redundant* (with respect to N), if there exist clauses $\mathcal{D}_1, \ldots, \mathcal{D}_k$ in $N_{\mathcal{C}}$ such that $\mathcal{D}_1, \ldots, \mathcal{D}_k, \mathcal{C}_1, \ldots, \mathcal{C}_n \models \mathcal{D}$. By $\mathcal{R}^{\succ}_{\mathcal{I}}(N)$ we denote the set of redundant inferences in Γ with respect to N. We emphasize that $\mathcal{R}^{\succ}_{\mathcal{I}}(N)$ will usually contain inferences whose premises are not in N.

Standard redundancy is mainly useful for inference systems with the reduction property, but the notion is well-defined for any inference system and ordering.

4.5. Lemma. *If* $N \subseteq N'$*, then* $\mathcal{R}^{\succ}_{\mathcal{F}}(N) \subseteq \mathcal{R}^{\succ}_{\mathcal{F}}(N')$*. Furthermore, if a clause* $\mathcal{C}$ *is redundant in* N*, then there exist non-redundant clauses* $\mathcal{C}_1, \ldots, \mathcal{C}_k$ *in* N*, such that* $\mathcal{C}_1, \ldots, \mathcal{C}_k \models \mathcal{C}$ *is valid and* $\mathcal{C} \succ \mathcal{C}_1, \ldots, \mathcal{C}_k$*. Consequently,* $\mathcal{R}^{\succ}_{\mathcal{F}}(N) \subseteq \mathcal{R}^{\succ}_{\mathcal{F}}(N \setminus \mathcal{R}^{\succ}_{\mathcal{F}}(N))$*, for all sets of clauses* N*.*

Proof. The first part follows immediately from the definition of redundancy. For the second part, suppose $\mathcal{C}$ is redundant in N. Let $N' = \mathcal{C}_1, \ldots, \mathcal{C}_k$ be a minimal subset of N (with respect to the multiset ordering $\succ_{mul}$), such that $\mathcal{C}_1, \ldots, \mathcal{C}_k \models \mathcal{C}$ and $\mathcal{C} \succ \mathcal{C}_j$, for all j. Then the clauses $\mathcal{C}_j$ are all non-redundant. □

The lemma indicates that a redundant clause $\mathcal{C}$ in N logically follows from $N_{\mathcal{C}} \setminus \mathcal{R}^{\succ}_{\mathcal{F}}(N)$.

4.6. Lemma. *If $N \subseteq N'$, then $\mathcal{R}^{\succ}_{\mathcal{I}}(N) \subseteq \mathcal{R}^{\succ}_{\mathcal{I}}(N')$. Moreover, $\mathcal{R}^{\succ}_{\mathcal{I}}(N) \subseteq \mathcal{R}^{\succ}_{\mathcal{I}}(N \setminus \mathcal{R}^{\succ}_{\mathcal{F}}(N))$, for all sets of clauses N.*

The proof uses Lemma 4.5.

4.7. Theorem. *$\mathcal{R}^{\succ}$ is a redundancy criterion.*

Proof. Lemmas 4.5 and 4.6 indicate that properties (R1) and (R2) are satisfied. In addition, the redundancy criterion $\mathcal{R}^{\succ}$ preserves inconsistency, as required by (R3). □

An inference with main premise $\mathcal{C}$ and conclusion $\mathcal{D}$ is called *reductive* with respect to an ordering $\succ$ if $\mathcal{C} \succ \mathcal{D}$. An inference system is called reductive if all its inferences are.

4.8. Theorem. *The standard redundancy criterion $\mathcal{R}^{\succ}$ is effective for any inference system that is reductive with respect to $\succ$.*

If an inference system Γ has the reduction property, the subset of its reductive inferences also satisfies the reduction property. Therefore one may ignore non-reductive inferences, and standard redundancy provides an effective criterion.

4.9. Theorem. *Let Γ be an inference system that satisfies the reduction property with respect to $\succ$, and let N be a set of clauses that is saturated up to redundancy with respect to Γ and the standard criterion $\mathcal{R}^{\succ}$. Then N is unsatisfiable if, and only if, it contains a contradiction.*

Proof. Suppose N is saturated and unsatisfiable, but contains no contradiction, and let M be the set $N \setminus \mathcal{R}^{\succ}_{\mathcal{F}}(N)$. Consider the interpretation I_M. If I_M is not a model of M, then M contains a minimal counterexample $\mathcal{C}$ for I_M. Since Γ has the reduction property there is an inference from M with main premise $\mathcal{C}$, side premises $\mathcal{C}_1, \ldots, \mathcal{C}_k$, and a conclusion $\mathcal{D}$, such that $\mathcal{D}$ is a smaller counterexample for I_M than $\mathcal{C}$ and the clauses $\mathcal{C}_i$ are true in I_M. By saturation, this inference is redundant. Thus, there are clauses $\mathcal{D}_1, \ldots, \mathcal{D}_m$ in $N_{\mathcal{C}}$, all smaller than $\mathcal{C}$, such that $\mathcal{D}$ logically follows from $\mathcal{C}_1, \ldots, \mathcal{C}_k, \mathcal{D}_1, \ldots, \mathcal{D}_m$. According to Lemma 4.5 we may assume that each clause $\mathcal{D}_j$ is non-redundant (and, therefore, is in M) and true in I_M (for $\mathcal{C}$ is the minimal counterexample for I_M). But this implies that $\mathcal{D}$ is true in I_M, which is a contradiction. In sum, I_M is a model of M, and also of N. □

To summarize, inference systems that satisfy the reduction property for counterexamples are refutationally complete and also compatible with application of the standard redundancy criterion. Standard redundancy, as will be seen below, is a powerful concept that justifies most, if not all, of the common simplification and

Ordered resolution with selection

$$\frac{C_1 \vee A_{11} \vee \ldots \vee A_{1k_1} \quad \ldots \quad C_n \vee A_{1n} \vee \ldots \vee A_{nk_n} \quad \neg A_1 \vee \ldots \vee \neg A_n \vee D}{C_1\sigma \vee \ldots \vee C_n\sigma \vee D\sigma}$$

where σ is a most general simultaneous solution of all unification problems $A_{i1} = \ldots = A_{ik_i} = A_i$, where $1 \leq i \leq n$, and
(i) either $A_1, \ldots, A_n$ are selected in D, or else nothing is selected in D, $n = 1$, and $A_1\sigma$ is maximal in $D\sigma$,
(ii) each atom $A_{ii}\sigma$ is strictly maximal with respect to $C_i\sigma$, and
(iii) no clause $C_i \vee A_{i1} \vee \ldots \vee A_{ik_i}$ contains a selected atom.

Figure 4: *Ordered Resolution for First-Order Standard Clauses* $\mathsf{O}_S^{\succ}$

deletion techniques used in refutational theorem provers. Next we discuss issues related to lifting these methods to clauses with variables.

4.3. A Simple Resolution Prover for First-Order Clauses

We will use an extended example to illustrate how the theoretical concepts of theorem proving derivations and redundancy can be applied to a realistic model of a resolution-based theorem prover with fundamental simplification techniques. The prover is applicable to *non-ground, standard* clauses, so that we also need to address the issue of lifting from the ground level to non-ground clauses.

Figure 4 defines a generalization of ordered resolution to standard first-order (i.e., non-ground) clauses via unification. If $\mathcal{O}$ is a set of clauses and if C is a clause, by $\mathsf{O}_S^{\succ}(\mathcal{O}, C)$ we denote the set of all inferences in $\mathsf{O}_S^{\succ}$ for which one of the premises is the clause C and the other premises are clauses in $\mathcal{O}$. We assume that $\succ$ is an admissible ordering on ground expressions that has been extended to non-ground expressions (atoms, literals, and clauses) as follows: $E \succ E'$ iff $E\sigma \succ E'\sigma$, for all ground substitutions σ. The extended ordering on non-ground expressions is only a partial (well-founded) ordering, even though the ordering is total on ground atoms. We also implicitly assume that different premises and the conclusion have no variables in common; variables are renamed if necessary. The selection function S is defined both for ground and non-ground clauses: $S(C)$ is a (possibly empty) sequence of negative atoms in C. The inference rule shown in Figure 4 coincides with the earlier inference rule in Figure 2 when all premises are ground. In that case, unifiability of atoms coincides with (syntactic) equality, and, since the ordering on ground expressions is total, the complement of the ordering $\succ$ is $\preceq$ and the complement of $\prec$ is $\succeq$.

The prover employs specific redundancy criteria. We say that a clause C *subsumes* a clause D (or that D *is subsumed by* C) if and only if there exists a substitution

Tautology deletion

$$\mathcal{N} \cup \{C\} \mid \mathcal{P} \mid \mathcal{O} \implies \mathcal{N} \mid \mathcal{P} \mid \mathcal{O} \qquad \text{if } C \text{ is a tautology}$$

Forward subsumption

$$\mathcal{N} \cup \{C\} \mid \mathcal{P} \mid \mathcal{O} \implies \mathcal{N} \mid \mathcal{P} \mid \mathcal{O} \qquad \text{if some clause in } \mathcal{P} \cup \mathcal{O} \text{ subsumes } C$$

Backward subsumption

$$\begin{array}{lcl} \mathcal{N} \mid \mathcal{P} \cup \{C\} \mid \mathcal{O} & \implies & \mathcal{N} \mid \mathcal{P} \mid \mathcal{O} \\ \mathcal{N} \mid \mathcal{P} \mid \mathcal{O} \cup \{C\} & \implies & \mathcal{N} \mid \mathcal{P} \mid \mathcal{O} \end{array}$$

$$\text{if some clause in } \mathcal{N} \text{ properly subsumes } C$$

Forward reduction

$$\mathcal{N} \cup \{C \vee L\} \mid \mathcal{P} \mid \mathcal{O} \implies \mathcal{N} \cup \{C\} \mid \mathcal{P} \mid \mathcal{O}$$

$$\text{if there is a clause } D \vee L' \text{ in } \mathcal{P} \cup \mathcal{O} \text{ such that } \overline{L} = L'\sigma \text{ and } D\sigma \subseteq C$$

Backward reduction

$$\begin{array}{lcl} \mathcal{N} \mid \mathcal{P} \cup \{C \vee L\} \mid \mathcal{O} & \implies & \mathcal{N} \mid \mathcal{P} \cup \{C\} \mid \mathcal{O} \\ \mathcal{N} \mid \mathcal{P} \mid \mathcal{O} \cup \{C \vee L\} & \implies & \mathcal{N} \mid \mathcal{P} \cup \{C\} \mid \mathcal{O} \end{array}$$

$$\text{if there is a clause } D \vee L' \text{ in } \mathcal{N} \text{ such that } \overline{L} = L'\sigma \text{ and } D\sigma \subseteq C$$

Clause processing

$$\mathcal{N} \cup \{C\} \mid \mathcal{P} \mid \mathcal{O} \implies \mathcal{N} \mid \mathcal{P} \cup \{C\} \mid \mathcal{O}$$

Inference computation

$$\emptyset \mid \mathcal{P} \cup \{C\} \mid \mathcal{O} \implies \mathcal{N} \mid \mathcal{P} \mid \mathcal{O} \cup \{C\} \qquad \text{where } \mathcal{N} = \mathsf{C}(\mathsf{O}_S^{\succ}(\mathcal{O}, C))$$

Figure 5: *The Resolution Prover* RP

σ such that $C\sigma$ is a sub-multiset of D. If C subsumes D, but not vice-versa, then C is said to *properly subsume* D. Subsumption defines a well-founded ordering on clauses. Two clauses C and D are said to be *variants* of each other if they mutually subsume each other.

The following resolution rule is of interest as it enables a subsequent subsumption:

Subsumption resolution

$$\frac{D \vee L \quad C \vee D\sigma \vee \overline{L}\sigma}{C \vee D\sigma}$$

This inference is used in combination with deletion, as the conclusion of this inference renders the second premise redundant. The inference *reduces* $C \vee D\sigma \vee \overline{L}\sigma$ to $C \vee D\sigma$.

Figure 5 depicts a binary relation $\implies$ that formalizes a resolution prover RP with tautology elimination, subsumption and subsumption resolution. The prover operates on triples $(\mathcal{N} \mid \mathcal{P} \mid \mathcal{O})$ of clause sets $\mathcal{N}$, $\mathcal{P}$, and $\mathcal{O}$ that represent a *state* of the theorem proving process in terms of newly derived resolvents, "processed" clauses, and "old" clauses. Initial states are of the form $(\mathcal{N} \mid \emptyset \mid \emptyset)$, where $\mathcal{N}$ is a *finite* set of possibly non-ground standard clauses.

The first five rules deal with redundancy elimination and simplification. Newly derived resolvents may be deleted if they are tautologies or are subsumed by processed or old clauses. In addition they may be simplified by reduction with old clauses. *Forward subsumption* allows us to remove a newly derived clause whenever it is subsumed by a processed or old clause, but *backward subsumption* is based on proper subsumption. A clause may be moved from the "new" set to the "processed" set at any time, but preferably after simplification. Once all new clauses have been processed, new resolution inferences are computed between some selected processed clause C and all old clauses, after which C becomes an old clause itself. The distinction between "processed" and "old" is useful for achieving fairness in inference computation.

Next we show that derivations by $\Longrightarrow$ represent theorem proving derivations on the sets of ground instances represented by the successive states. We denote by $G(C)$ the set of all *ground instances* $C\sigma$ of a clause C. G is extended to sets of clauses by taking the union of the sets of ground instances of the clauses in the set. If $\mathcal{S}_i = (\mathcal{N}_i \mid \mathcal{P}_i \mid \mathcal{O}_i)$ is a state, we sometimes, ambiguously, identify $\mathcal{S}_i$ with the set $\mathcal{N}_i \cup \mathcal{P}_i \cup \mathcal{O}_i$. In particular, by $G(\mathcal{S}_i)$ we denote the set $G(\mathcal{N}_i) \cup G(\mathcal{P}_i) \cup G(\mathcal{O}_i)$ of all ground instances of clauses present in state $\mathcal{S}_i$.

Consider a derivation

$$\mathcal{N}_0 \mid \emptyset \mid \emptyset \Longrightarrow \mathcal{N}_1 \mid \mathcal{P}_1 \mid \mathcal{O}_1 \Longrightarrow \mathcal{N}_2 \mid \mathcal{P}_2 \mid \mathcal{O}_2 \Longrightarrow \ldots$$

on states $\mathcal{S}_i = (\mathcal{N}_i \mid \mathcal{P}_i \mid \mathcal{O}_i)$, where $\mathcal{N}_0$ is a finite set of clauses and $\mathcal{P}_0 = \mathcal{O}_0 = \emptyset$. Let N_i be an abbreviation for the sets of ground instances $G(\mathcal{S}_i)$. We will first show that the sequence $N_0, N_1, N_2, \ldots$ represents a theorem proving derivation $\rhd$ based on some extension (in the sense of Section 4.1) of ordered resolution $O_S^{\succ}$ with standard redundancy $\mathcal{R}^{\succ}$. The inferences on which the derivations in RP are based include those in $O_S^{\succ}$. But there are also other inferences applied, such as subsumption resolution. Moreover, even if $\mathcal{S}_{i+1}$ results from $\mathcal{S}_i$ by a step of inference computation in (the non-ground version of) $O_S^{\succ}$, the clauses in $N_{i+1} \setminus N_i$ might not all be representable as conclusions of inferences in (the ground version) of $O_S^{\succ}$. For instance, the inference

$$\frac{p(f(x,a)) \quad \neg p(f(y,z)) \vee \neg p(f(z,y))}{\neg p(f(a,x))}$$

will be in $O_S^{\succ}$ for many orderings, if nothing is selected in the second premise. Its ground instance

$$\frac{p(f(b,a)) \quad \neg p(f(b,a)) \vee \neg p(f(a,b))}{\neg p(f(a,b))}$$

however, is not in $O_S^{\succ}$, if either $p(f(a,b)) \succ p(f(b,a))$, or else if $\neg p(f(b,a))$, but not $\neg p(f(a,b))$, is selected. Selection is generally not compatible with instantiation.

4.10. Lemma. *If $\mathcal{S} \Longrightarrow \mathcal{S}'$ then $G(\mathcal{S}) \rhd^* G(\mathcal{S}')$, with $\rhd$ based on some extension of $\mathsf{O}_S^{\succ}$ and $\mathcal{R}$.*

PROOF. We proceed by a case analysis over the definition of $\Longrightarrow$.

Tautology elimination: Suppose $\mathcal{S} = (\mathcal{N} \cup \{C\} \mid \mathcal{P} \mid \mathcal{O})$ and $\mathcal{S}' = (\mathcal{N} \mid \mathcal{P} \mid \mathcal{O})$, where C is a tautology. Then any instance of C is also a tautology, and hence redundant. In other words, $G(\mathcal{S}) \setminus G(\mathcal{S}') \subseteq \mathcal{R}_{\mathcal{F}}(G(\mathcal{S}))$.

Forward subsumption: Let $\mathcal{S} = (\mathcal{N} \cup \{C\} \mid \mathcal{P} \mid \mathcal{O})$ and $\mathcal{S}' = (\mathcal{N} \mid \mathcal{P} \mid \mathcal{O})$ with C subsumed by some D in $\mathcal{P} \cup \mathcal{O}$, that is, $D\sigma \subseteq C$, for some substitution σ. If $D\sigma = C$, then $G(C) \subseteq G(D)$, hence $G(\mathcal{S}) = G(\mathcal{S}')$. Otherwise, $D\sigma \subset C$ and $G(C) \subseteq \mathcal{R}_{\mathcal{F}}(\{G(D)\})$. In both cases, $G(\mathcal{S}) \rhd G(\mathcal{S}')$.

Backward subsumption: similar.

Forward reduction: Let $\mathcal{S} = (\mathcal{N} \cup \{C \vee L\} \mid \mathcal{P} \mid \mathcal{O})$ and $\mathcal{S}' = (\mathcal{N} \cup \{C\} \mid \mathcal{P} \mid \mathcal{O})$, with $D \vee L'$ in $\mathcal{P} \cup \mathcal{O}$ such that $\overline{L} = L'\sigma$ and $D\sigma \subseteq C$. Reduction can be viewed as a two-step process in which first C is derived by subsumption resolution and then $C \vee L$ is deleted by subsumption. Let $\mathcal{S}'' = (\mathcal{N} \cup \{C \vee L, C\} \mid \mathcal{P} \mid \mathcal{O})$. Clearly, $D \vee L', C \vee L \models C$, and therefore $G(\mathcal{S}) \rhd G(\mathcal{S}'')$ by a step of deduction in an extended inference system. Moreover, $C \vee L$ is properly subsumed by C so that $G(C \vee L) \subseteq \mathcal{R}_{\mathcal{F}}(G(\mathcal{S}''))$, and $G(\mathcal{S}'') \rhd G(\mathcal{S}')$ by a step of deletion. Altogether, $G(\mathcal{S}) \rhd^* G(\mathcal{S}')$.

Backward reduction: similar.

Clause processing: Here the set of ground clauses represented by the proof state is not changed so that $G(\mathcal{S}) \rhd G(\mathcal{S}')$ is trivially true.

Inference selection: Inferences in $\mathsf{O}_S^{\succ}$ produces logical consequences of clauses in $\mathcal{S}$, so that the ground instances of the new clauses in $\mathcal{N}$ can be viewed as derived in an appropriately extended inference system. Therefore $G(\mathcal{S}) \rhd G(\mathcal{S}')$. □

Since the preceding lemma does not reveal much information about the inference system that is the basis of the derivations on the ground level, the real work of showing refutational completeness of RP will be concerned with issues of fairness. More precisely, we have to identify criteria on the level of proof states that will ensure fairness of the corresponding ground derivations $N_0, N_1, \ldots$ with respect to $\mathsf{O}_{S'}^{\succ}$ and standard redundancy $\mathcal{R}$, for certain selection functions S' derived from S.

Let $\mathcal{S}_\infty$ be the triple $(\mathcal{N}_\infty \mid \mathcal{P}_\infty \mid \mathcal{O}_\infty)$. A sequence of states $\mathcal{S}_i$ is called *fair* if $\mathcal{N}_\infty = \mathcal{P}_\infty = \emptyset$. That is, we require that each clause will be eventually processed and that each processed clause will eventually be selected for inference computation.[4]

4.11. LEMMA. *For any fair sequence of states $\mathcal{S}_1 \Longrightarrow \mathcal{S}_2 \Longrightarrow \cdots$ we have $G(\mathcal{S}_\infty) \supseteq N_\infty \setminus \mathcal{R}(N_\infty)$, where N_∞ denotes the limit of the ground derivation $N_i = G(\mathcal{S}_i)$ represented by the sequence of states.*

PROOF. Assume that C is a ground clause in $N_\infty \setminus \mathcal{R}(N_\infty)$, in particular, $C \in N_j$, for all $j \geq i$, with some $i \geq 0$. Being non-redundant, C is not a tautology. If D is

[4]Fairness can be ensured, for instance, by selecting clauses of minimal index for inference computation, where index refers to a weight function that is monotonically increasing both in the size of a clause and the number of the state in which it was produced. The latter component is needed to avoid potentially unfair selection strategies when some clause is generated infinitely often and a larger clause is never selected for inference computation.

a clause in some of the $\mathcal{N}_j$, $j \geq i$, such that C is an instance of D, then, as the sequence is fair, D will be deleted from $\mathcal{N}_{l-1}$ at some subsequent step $l-1 \geq j$. From Lemma 4.10 and from the definition of RP this can only be the case if C is also instance of a clause D' in some $\mathcal{P}_l \cup \mathcal{O}_l$. (Since C is not redundant, D cannot be deleted from $\mathcal{N}$ by forward reduction as the reduct that would be added would properly subsume D. Therefore, either D is subsumed by some D' in $\mathcal{P}_l \cup \mathcal{O}_l$, or else D is moved from $\mathcal{N}_{l-1}$ to $\mathcal{P}_l$ by a step of clause processing.) As $\mathcal{P}_\infty$ is empty, if D' is in $\mathcal{P}_l$, it will eventually also be in $\mathcal{O}_{l'}$, for some $l' > l$. Then, D' will be in each of the $\mathcal{O}_k$, with $k \geq l'$. (Again we make use of the fact that since C is not redundant, D' cannot be deleted by backward reduction or backward subsumption.) Thus, we have shown that there is a clause D' in $\mathcal{S}_\infty$ having C as its ground instance. □

It is a classical result that (unrestricted) resolution inferences can be lifted in that any inference from ground instances of clauses can be obtained as instances of inferences from the clauses themselves. In our setting a particular technical problem arises in that the selection function S need not be compatible with substitution. If a ground clause is an instance of two different non-ground clauses, which of the two possibly different selections should be inherited by the instance? Fortunately, these ambiguities are not critical. Suppose we have a selection function S and a set M of clauses with ground instances $K = G(M)$. Then let S_M denote an arbitrary new selection function for which (i) if C is in K, then $S_M(C) = S(D)\sigma$, for some D in M such that $C = D\sigma$; and (ii) $S_M(C) = S(C)$, if C is not in K. In other words, for any clause C in K, $S_M(C)$ needs to coincide with $S(D)$ for at least one clause D in M that has C as one of its ground instances. Depending on the choice of D for C, there may be different such functions S_M. Any choice of S_M gives us the required degree of compatibility between selection on the ground and non-ground level, respectively, as formalized by this lifting lemma:

4.12. Lemma (Lifting Lemma). *Let M be a set of clauses and $K = G(M)$. If*

$$\frac{C_1 \quad \ldots C_n \quad C_0}{C}$$

is an inference in $\mathsf{O}^{\succ}_{S_M}(K)$ then there exist clauses C'_i in M, a clause C', and a ground substitution σ such that

$$\frac{C'_1 \quad \ldots C'_n \quad C'_0}{C'}$$

is an inference in $\mathsf{O}^{\succ}_{S}(M)$, $C_i = C'_i\sigma$, and $C = C'\sigma$.

Proof. We choose for the premises those clauses C'_i in M which were used for defining $S_M(C_i)$ as the multiset of literals corresponding to the multiset selected by $S(C'_i)$. In that case a resolution inference from the C'_i exists and satisfies the restrictions about $\succ$ and S. □

4.13. THEOREM. *If $\mathcal{S}_0 \Longrightarrow \mathcal{S}_1 \Longrightarrow \ldots$ is a fair derivation, then $\mathcal{S}_0$ is unsatisfiable if and only if $\mathcal{S}_\infty$ contains the empty clause.*

PROOF. Let again N_i denote $G(\mathcal{S}_i)$. From Lemma 4.10 we may infer that the sequence of sets N_i is a theorem proving derivation for which, by Lemma 4.11, we have $N_\infty \setminus \mathcal{R}(N_\infty) \subseteq \mathcal{O}_\infty$. We will show that N_∞ is saturated with respect to $\mathsf{O}^{\succ}_{S_{\mathcal{O}_\infty}}$, where $S_{\mathcal{O}_\infty}$ is an arbitrary selection function derived from S and $\mathcal{O}_\infty$. Let γ be an inference in $\mathsf{O}^{\succ}_{S_{\mathcal{O}_\infty}}$ from non-redundant premises in $G(\mathcal{O}_\infty)$ having conclusion C. By the lifting lemma there is an inference in $\mathsf{O}^{\succ}_S$ from clauses C'_i in $\mathcal{O}_\infty$, $1 \leq i \leq n$, such that the conclusion C' has C as a ground instance. All inferences by $\mathsf{O}^{\succ}_S$ from the clauses C'_i will have been considered when the "youngest" clause C'_i becomes old. Therefore γ is redundant in some $G(\mathcal{S}_j)$. By Lemma 4.10, γ is also redundant in $G(\mathcal{O}_\infty) = G(\mathcal{S}_\infty)$. Therefore, N_∞ is saturated up to redundancy with respect to $\mathsf{O}^{\succ}_{S_{\mathcal{O}_\infty}}$ and $\mathcal{R}$, and we may infer that either N_0 is satisfiable or else N_∞ (and hence $\mathcal{S}_\infty$) contains the empty clause. $\mathcal{S}_0$ is satisfiable if and only in N_0 is satisfiable, from which the assertion follows. □

Our investigation of the properties of RP shows that there are considerable technical complications in proving that refutational completeness is preserved by forward subsumption and strict backward subsumption. But if one admits a more liberal notion of backward subsumption, a ground instance $C = D\sigma = D'\sigma'$ may persist, even though neither D nor D' persists on the non-ground level. Thus more complex criteria have to be employed and checked by the prover to ensure fairness.

5. General Resolution

Ordered resolution with selection is a versatile inference system that can be adapted to specific applications not only by adjusting the two parameters of ordering and selection function, but also by exploiting redundancy criteria. These advantages are especially useful for the inference rules on general clauses, a generalization we proceed to describe next. General clauses are multisets of arbitrary quantifier-free formulae, denoting the disjunction of their elements. In this and the subsequent sections all expressions are assumed to be ground, *unless indicated otherwise*. Lifting the new inference systems to non-ground clauses can be done in a way similar to what we have described for ordered resolution on standard clauses in the context of the prover RP in Section 4.3. In Section 9 we will briefly describe refined lifting methods involving, for instance, constrained clauses.

5.1. Selection Functions

First we need to generalize the notion of a selection function. Intuitively, if $\mathcal{C}$ is the minimal counterexample to a candidate model I then one of the sequences selected by $S(\mathcal{C})$ should be true in I, so that $\mathcal{C}$ can be reduced to a smaller counterexample by resolving on selected atoms only.

A (general) *selection function* is a mapping S that assigns to each general clause $\mathcal{C}$ a (possibly empty) set $S(\mathcal{C})$ of nonempty sequences of (distinct) atoms in $\mathcal{C}$ such that either $S(\mathcal{C})$ is empty or else, for all interpretations I in which $\mathcal{C}$ is false, there exists a sequence $A_1, \ldots, A_k$ in $S(\mathcal{C})$, all atoms of which are true in I. A sequence $A_1, \ldots, A_k$ in $S(\mathcal{C})$ is said to be *selected* (by S). Sometimes the atoms A_i are also called *selected*, especially if a sequence in $S(\mathcal{C})$ consists of a single atom. If $S(\mathcal{C})$ is empty, the clause $\mathcal{C}$ contains no selected atom.

For example, possible choices for selection in the clause $\mathcal{C} = (\neg A \wedge \neg B, \neg A', B')$ are $S_1(\mathcal{C}) = \{(A, A'), (B, A')\}$, $S_2(\mathcal{C}) = \{A, B\}$, and $S_3(\mathcal{C}) = \{A'\}$.

Verifying that a set $S(\mathcal{C})$ satisfies the required conditions for given S and $\mathcal{C}$ is a computationally hard problem in general, though sufficient (syntactic) criteria can be specified in special cases. For instance, if C is a standard clause, then those atoms that occur in negative literals of C are precisely the ones that may be selected, and any set of sequences of negative atoms forms a legal selection. For general clauses, selection can be based on a suitable notion of the *polarity* of a subformula in an expression. We say that a subformula of F' in $E[F']$ is *positive* (respectively, *negative*), if $E[F'/\top]$ (respectively, $E[F'/\bot]$) is a tautology. Thus, if F' is positive (respectively, negative) in E, then F' (respectively, $\neg F'$) logically implies E.

For example, in a disjunction $A \vee B$ both A and B are positive, whereas in a conjunction $A \wedge B$ the two subformulas A and B are neither positive nor negative. A subformula may occur both positively and negatively (e.g., A in $A \vee \neg A$ or $A \equiv A$), in which case the formula is a tautology. For standard clauses $\neg A_1 \vee \ldots \vee \neg A_m \vee B_1 \vee \ldots \vee B_n$ we obtain the usual notion of polarity in that all atoms A_i are negative and all atoms B_j are positive. It is safe to select any sequence of negative atoms in a general clause, as a negative atom cannot be false in any interpretation in which the clause is false.

Determining whether an atom A is positive or negative in a formula E requires one to check whether $E[A/\top]$ or $E[A/\bot]$ is a tautology, which is again a computationally hard problem. But syntactic criteria allow one to identify certain positive and negative occurrences of atoms in a clause in linear time (in the size of the given clause).

5.1. Proposition. *(i) F is a positive subformula of F.*
(ii) If $\neg G$ is a positive (respectively, negative) subformula of F, then G is a negative (respectively, positive) subformula of F.
(iii) If $G \vee H$ is a positive subformula of F, then G and H are both positive subformulas of F.
(iv) If $G \wedge H$ is a negative subformula of F, then G and H are both negative subformulas of F.
(v) If $G \supset H$ is a positive subformula of F, then G is a negative subformula and H is a positive subformula of F.
(vi) If $G \supset \bot$ is a negative subformula of F, then G is a positive subformula of F.
(vii) F is positive in a clause $\mathcal{C}$ if it is an element of $\mathcal{C}$.

5.2. General Ordered Resolution

Let $\succ$ be an ordering and S be a (general) selection function. The inference system $\mathsf{O}_S^{\succ}$ of *general ordered resolution* is depicted in Figure 6. In the case of self-resolution, there is only a main premise, no side premises. In ordered resolution, the last premise is the main premise, while the other premises are the side premises. We occasionally omit the subscript and/or superscript in $\mathsf{O}_S^{\succ}$ if the ordering $\succ$ or the selection function S are clear from the context. In an ordered resolution inference either all selected atoms or, if there are no selected atoms, the maximal atom in the main premise are resolved, and the side premises must not contain any selected atoms.

Note that the order among the two premises is significant. For example, resolution on A in $A \vee B$ and $A \supset C$ yields the resolvent $(\bot \vee B), (\top \supset C)$, which is logically equivalent to B, C. If we exchange the premises, we obtain $(\bot \supset C), (\top \vee B)$, a tautology.

Let us consider two examples.

(1) $A \equiv B$ [input]
(2) $\neg A \vee \neg B$ [input]
(3) $A \vee B$ [input]
(4) $(\bot \vee B), (\top \equiv B)$ [resolving on A in (3) and (1)]
(5) $(\bot \equiv B), (\neg\top \vee \neg B)$ [resolving on A in (1) and (2)]
(6) $(\bot \vee \bot), (\top \equiv \bot), (\bot \equiv \top), (\neg\top \vee \neg\top)$
[resolving on B in (4) and (5)]

Clause (6) is a contradiction as each disjunct simplifies to $\bot$. Hence, by soundness there is no interpretation in which all three input formulas are true.

A different approach is to repeatedly apply self-resolution to the conjunction of the given (finitely many) input formulas until all atoms have been eliminated:

(1) $(A \equiv B) \wedge (\neg A \vee \neg B) \wedge (A \vee B)$ [input]
(2) $[(\bot \equiv B) \wedge (\neg\bot \vee \neg B) \wedge (\bot \vee B)]$
, $[(\top \equiv B) \wedge (\neg\top \vee \neg B) \wedge (\top \vee B)]$ [resolving on A in (1)]
(3) $[(\bot \equiv \bot) \wedge (\neg\bot \vee \neg\bot) \wedge (\bot \vee \bot)]$
, $[(\top \equiv \bot) \wedge (\neg\top \vee \neg\bot) \wedge (\top \vee \bot)]$
, $[(\bot \equiv \top) \wedge (\neg\bot \vee \neg\top) \wedge (\bot \vee \top)]$
, $[(\top \equiv \top) \wedge (\neg\top \vee \neg\top) \wedge (\top \vee \top)]$ [resolving on B in (2)]

(3) is a contradiction as each disjunct contains a false conjunct.

The examples illustrate two extreme cases in the wide spectrum of possible resolution strategies—ranging from local strategies where replacement of (atomic) subformulas is confined to single standard clauses, to global ones in which the entire state of the theorem proving process is modified non-locally at each step.

General ordered resolution with selection

$$\frac{\mathcal{C}_1(A_1) \quad \ldots \quad \mathcal{C}_n(A_n) \quad \mathcal{D}(A_1,\ldots,A_n)}{\mathcal{C}_1(\bot)\,,\ldots,\mathcal{C}_n(\bot)\,,\mathcal{D}(\top,\ldots,\top)}$$

where (i) either $A_1,\ldots,A_n$ is selected by S in $\mathcal{D}$, or else $S(\mathcal{D})$ is empty, $n=1$, and A_1 is maximal in $\mathcal{D}$, (ii) each atom A_i is maximal in $\mathcal{C}_i$, and (iii) no clause $\mathcal{C}_i$ contains a selected atom.

Ordered self-resolution

$$\frac{\mathcal{D}(A)}{\mathcal{D}(\bot)\,,\mathcal{D}(\top)}$$

where (i) the atom A is maximal in the premise, and (ii) the premise contains no selected atom.

Figure 6: *General Ordered Resolution* $\mathsf{O}_S^{\succ}$

5.3. Refutational Completeness

We establish the refutational completeness of general ordered resolution by showing that the calculus has the reduction property for counterexamples. We essentially use the model functor I from Definition 3.14, but applied to general clauses and with an additional restriction on productive clauses.

Let N be a set of general clauses and $\succ$ be an admissible ordering. We use induction on $\succ$ to define, for each clause $\mathcal{C}$, a Herbrand interpretation $I_\mathcal{C}$ and a set $\varepsilon_\mathcal{C}$ as follows.

5.2. Definition. Take $I_\mathcal{C}$ to be the set $\bigcup_{\mathcal{C}\succ\mathcal{D}}\varepsilon_\mathcal{D}$. If A is the maximal atomic formula of a clause $\mathcal{C}$ in N, then $\varepsilon_\mathcal{C}=\{A\}$ if (i) $A\notin I_\mathcal{C}$, (ii) $\mathcal{C}$ is false in $I_\mathcal{C}$, but true in $I_\mathcal{C}\cup\{A\}$, and (iii) $\mathcal{C}$ contains no selected atom; otherwise, $\varepsilon_\mathcal{C}$ is the empty set.

We say that $\mathcal{C}$ *produces* A, and call $\mathcal{C}$ *productive*, if $\varepsilon_\mathcal{C}=\{A\}$. Finally, I_N is defined as the Herbrand interpretation $\bigcup_{\mathcal{C}\in N}\varepsilon_\mathcal{C}$. Whenever we wish to emphasize the dependency on the ordering we write $I_N^{\succ}$. The additional restriction on productive clauses compared to Definition 3.14 is that an atom A is produced only if the clause becomes true in the extended interpretation. Hence, a clause such as $A\wedge\bot$ cannot be productive, and, typically, self-resolution (on A) has to be applied to split the clause into two sub-cases.

The inferences in $\mathsf{O}_S^{\succ}$ are reductive in the following sense.

5.3. Lemma. *Let $\succ$ be an admissible clause ordering and S be any selection function. Then the conclusion of any inference in $\mathsf{O}_S^{\succ}$ is smaller than the main premise.*

Note that the lemma would not hold for resolution of selected non-maximal atoms if conclusions were written as disjunctions, instead of as multisets.

5.4. THEOREM. *Let $\succ$ be an admissible clause ordering, N be a set of clauses not containing a contradiction, and $\mathcal{C}$ be a minimal counterexample in N for I_N. Then there exists an inference in $\mathsf{O}_S^{\succ}$ with main premise $\mathcal{C}$, the conclusion of which is a smaller counterexample for I_N than $\mathcal{C}$, and the side premises of which, in the case of ordered resolution, are productive clauses.*

PROOF. Let I be an abbreviation for I_N and $\mathcal{D}$ be the minimal counterexample in N for I. The clause $\mathcal{D}$ is not a contradiction and cannot be productive (as it is false in I). We distinguish two cases.

(i) If $\mathcal{D}$ contains no selected atoms, let A be the maximal atom in $\mathcal{D}$ (with respect to $\succ$).

(i.1) Suppose A is false in I. Then there is a self-resolution inference with premise $\mathcal{D}$ and resolvent $\mathcal{D}' = \mathcal{D}(\bot)\,, \mathcal{D}(\top)$, where clearly $\mathcal{D} \succ \mathcal{D}'$. Moreover, $\mathcal{D}(\top)$ is false in I, for otherwise $\mathcal{D}$ would produce A. Since A is false in I, the clause $\mathcal{D}(\bot)$ has the same truth value as $\mathcal{D}$ in I, i.e., is false. In short, the clause $\mathcal{D}'$ is a smaller counterexample for I than $\mathcal{D}$.

(i.2) If A is true in I, it must be produced by some clause $\mathcal{C}$ and the clause $\mathcal{C}(\bot)$ is therefore false in I. Since $\mathcal{D}(\top)$ is also false in I, the non-clausal resolvent $\mathcal{D}' = \mathcal{C}(\bot)\,, \mathcal{D}(\top)$ of $\mathcal{C}$ and $\mathcal{D}$ is a smaller counterexample for I than $\mathcal{D}$.

(ii) If $S(\mathcal{D})$ is nonempty, then by the properties of a selection function it must contain a sequence of atoms $A_1, \ldots, A_n$, all of which are true in I (for $\mathcal{D}$ is false in I). Each atom A_i is produced by some clause $\mathcal{C}_i$ in N. Using these clauses as side premises and $\mathcal{D}$ as main premise, we obtain an ordered resolution inference with resolvent

$$\mathcal{D}' = \mathcal{C}_1(\bot)\,, \ldots, \mathcal{C}_n(\bot)\,, \mathcal{D}(\top, \ldots, \top).$$

It can easily be shown that $\mathcal{D}'$ is a smaller counterexample for I than $\mathcal{D}$. □

The theorem indicates that general ordered resolution with selection has the reduction property for counterexamples, and hence is refutationally complete.

5.5. THEOREM (Refutational completeness). *Let $\succ$ be an admissible ordering and S be a selection function. If N is saturated up to standard redundancy under $\mathsf{O}_S^{\succ}$, then N is unsatisfiable if and only if it contains a contradiction.*

PROOF. If N contains a contradiction, then it is unsatisfiable. If N contains no contradiction, we may use Theorem 4.9 in combination with Theorem 5.4, to infer that it has a model. □

5.4. Applications of Standard Redundancy

The above completeness result, Theorem 5.5, covers a wide range of resolution-based theorem proving strategies. We next show how further refinements may be obtained by application of standard redundancy. From now on "redundancy" will refer to "standard redundancy," unless stated otherwise. The following observation will useful for reasoning about redundancy.

5.6. Proposition. *An inference by ordered resolution with side premises* $\mathcal{C}_1, \ldots, \mathcal{C}_n$*, main premise* $\mathcal{C}$*, resolved atoms* $A_1, \ldots, A_n$*, and conclusion* $\mathcal{C}'$ *is redundant in* N *if there exist clauses* $\mathcal{D}_1, \ldots, \mathcal{D}_k$ *in* N *such that* $\mathcal{C} \succ \mathcal{D}_1, \ldots, \mathcal{D}_k$ *and*

$$\mathcal{D}_1, \ldots, \mathcal{D}_k, \mathcal{C}_1, \ldots, \mathcal{C}_n, A_1, \ldots, A_n \models \mathcal{C}'.$$

Proof. We show that under the given assumptions the clause $\mathcal{C}'$ is a logical consequence of $\mathcal{D}_1, \ldots, \mathcal{D}_k, \mathcal{C}_1, \ldots, \mathcal{C}_n$. Suppose all clauses $\mathcal{D}_i$ and $\mathcal{C}_j$ are true in an interpretation I. Since $\mathcal{C}'$ is of the form $\mathcal{C}_1[A_1/\bot], \ldots, \mathcal{C}_n[A_n/\bot], \mathcal{D}'$, it is true in I if at least one of the atoms A_i is false in I. On the other hand, if all atoms A_i are true in I, then by the assumptions, $\mathcal{C}'$ is also true in I. □

5.4.1. Polarity-Based Restrictions

An inference is trivially redundant if its conclusion is a tautology. Some such redundant inferences can be detected by an analysis of the polarity of resolved atoms.

5.7. Proposition. *An inference by ordered resolution is redundant if (i) the main premise contains a positive occurrence of some resolved atom* A_i*, or (ii) some side premise* $\mathcal{C}_i$ *contains a negative occurrence of* A_i*, or (iii) some side premise* $\mathcal{C}_i$ *contains a positive occurrence of a resolved atom* A_j*, where* $A_j \neq A_i$*.*

Proof. Let $\mathcal{C}_1(A_1), \ldots, \mathcal{C}_n(A_n)$ be the side premises, and $\mathcal{D}(A_1, \ldots, A_n)$ the main premise of an ordered resolution inference that resolves $A_1, \ldots, A_n$. The resolvent is of the form $\mathcal{D}' = \mathcal{C}_1(\bot), \ldots, \mathcal{C}_n(\bot), \mathcal{D}(\top, \ldots, \top)$. If an atom A_i occurs positively in $\mathcal{D}$ then $\mathcal{D}(\top, \ldots, \top)$, and hence $\mathcal{D}'$, is a tautology. If an atom A_i occurs negatively in $\mathcal{C}_i(A_i)$, then $\mathcal{C}_i(\bot)$, and hence $\mathcal{D}'$, is a tautology. If an atom A_j, with $A_j \neq A_i$, occurs positively in $\mathcal{C}_i$, then $\mathcal{C}_i[A_j/\top]$ is a tautology, as is $\mathcal{C}_i[A_i/\bot, A_j/\top]$. Therefore A_j entails $\mathcal{C}_i[A_i/\bot]$ and hence the conclusion $\mathcal{D}'$. By Proposition 5.6, the inference is again redundant. □

In a similar way one can prove:

5.8. Proposition. *An inference by self-resolution is redundant if the resolved atom occurs positively or negatively in the premise.*

Propositions 5.7 and 5.8 suggest additional (polarity) constraints that may be safely attached to inferences in $\mathsf{O}_S^{\succ}$.

Let us briefly remark that Manna and Waldinger [1980] and Murray [1982] proposed a different restriction for general resolution, where the resolved atom A is required to occur positively in the positive premise and negatively in the negative premise. This requires a different notion of polarity, according to which each subformula is positive or negative (or both). For instance, A and B are considered to be positive in $A \wedge B$. These polarity constraints are not compatible with ordering constraints or selection and the combination yields an incomplete calculus. For example, take the formulas $A \wedge B$ and $\neg B$ and suppose $A \succ B$ and A and B are considered positive in $A \wedge B$. The inference

$$\frac{A \wedge B \quad \neg B}{(A \wedge \bot)\,, \neg\top}$$

satisfies the polarity constraint: B is positive in the first and negative in the second premise; but not the ordering constraint: B is not maximal in the first premise. (The resolvent, by the way, is a contradictory formula, but not a contradiction in our sense.) On the other hand, the inference

$$\frac{A \wedge B \quad A \wedge B}{(\bot \wedge B)\,,(\top \wedge B)}$$

satisfies the ordering constraint, but not the polarity constraint, as A occurs only positively in both premises. The resolvent of this inference is equivalent to B; and another resolution step with $\neg B$ yields a contradiction. There is no resolution inference from $A \wedge B$ and $\neg B$ that satisfies both polarity and ordering constraints. In other words, the simultaneous application of both kinds of constraints renders non-clausal resolution incomplete.

Theorem 5.7 already holds for a weaker form of negative polarity in which an atom A is considered negative in $\mathcal{C}(A)$ if $\mathcal{C}$ implies $\mathcal{C}(\bot)$.

5.4.2. Positive Resolution

A general clause is called *positive*, if it is false in the *empty Herbrand interpretation* $I_\bot$ (in which all atoms are false). In a positive clause no atoms can be selected, so that $S(\mathcal{C})$ must be the empty set, for any selection function S. Conversely, if a clause is not positive, simply selecting all its non-positive atoms yields a legal selection function, for if a non-positive clause C is false in an interpretation I then I must contain an atom A that is non-positive in C. Also note that a positive clause cannot contain a negative occurrence of an atom, though the converse is not true in general.

Let S be a selection function such that $S(\mathcal{C})$ is the empty set, if $\mathcal{C}$ is positive, and $S(\mathcal{C})$ is the set of all sequences (of length one consisting) of a single non-positive atom in $\mathcal{C}$, otherwise. General ordered resolution with such a selection function specializes to positive resolution as shown in Figure 7. The polarity constraints imposed on the inferences derive their justification from Propositions 5.7 and 5.8. As an immediate consequence of these propositions and Theorem 5.5 we get:

5.9. THEOREM. *If N is saturated up to redundancy under $\mathsf{GP}^{\succ}$, then N is unsatisfiable if and only if it contains a contradiction.*

In the case of standard clauses, the main premise of a positive resolution inference must be a non-positive clause. In the general case this restriction is too strong. For

General positive ordered resolution

$$\frac{\mathcal{C}(A) \quad \mathcal{D}(A)}{\mathcal{C}(\bot), \mathcal{D}(\top)}$$

where (i) the atom A is maximal in $\mathcal{C}$, (ii) the clause $\mathcal{C}$ is positive, and (iii) the atom A is non-positive in $\mathcal{D}$.

General positive ordered self-resolution

$$\frac{\mathcal{D}(A)}{\mathcal{D}(\bot), \mathcal{D}(\top)}$$

where (i) the atom A is maximal in $\mathcal{D}$, and (ii) the clause $\mathcal{D}$ is positive.

Figure 7: *General positive resolution* $\mathsf{GP}^{\succ}$

example, there is no such inference from an inconsistent set consisting of the three atoms A, B, and C, and the positive clause $(\neg A \wedge C, \neg B \wedge C)$, with A the maximal atom.

5.4.3. Partial Replacement Strategies

Resolution inferences essentially represent a case analysis on the truth values of certain atoms. We have formulated the general inferences in such a way that all occurrences of these atoms are replaced simultaneously. The following propositions show that a more selective, "partial" replacement is also possible.

5.10. Proposition. *Let*

$$\frac{\mathcal{C}_1(A_1) \ \ \ldots \ \ \mathcal{C}_n(A_n) \quad \mathcal{D}(A_1, \ldots, A_n)}{\mathcal{C}_1(\bot), \ldots, \mathcal{C}_n(\bot), \mathcal{D}(\top, \ldots, \top)}$$

be a general ordered resolution inference, $\{i_1, \ldots, i_k\}$ *be any non-empty subset of* $\{1, \ldots, n\}$*, and* P *be any non-empty set of occurrences of the atoms* A_{i_j} *in* $\mathcal{D}$*. If the "partial conclusion"*

$$\mathcal{C} = \mathcal{C}_{i_1}(\bot), \ldots, \mathcal{C}_{i_k}(\bot), \mathcal{D}[\top]_P$$

logically follows from $\mathcal{C}_1, \ldots, \mathcal{C}_n$ *and clauses in* N *smaller than* $\mathcal{D}$*, then the above inference is redundant in* N*.*

Proof. Let $\Gamma_1, \ldots, \Gamma_m$ be clauses in N such that

$$\Gamma_1, \ldots, \Gamma_m, \mathcal{C}_1, \ldots, \mathcal{C}_n \models \mathcal{C}.$$

Then the conclusion of the ordered resolution inference is implied by the clauses

$$\Gamma_1, \ldots, \Gamma_m, A_1, \ldots, A_n, \mathcal{C}_1, \ldots, \mathcal{C}_n,$$

so that the inference is redundant by Proposition 5.6. □

A resolution inference in which atoms $A_1, \ldots, A_n$ are simultaneously resolved can be implemented by a sequence of "partial" inferences, each of which resolves only some of the atoms (with the corresponding side premises).[5] Partial inferences are reductive and the redundancy of a partial inference therefore implies the redundancy of the original inference. Thus, the above proposition indicates that one may don't-care non-deterministically choose which of the selected atoms to resolve and also pick the positions in the main premises at which to resolve.

We obtain a corresponding result for self-resolution.

5.11. Proposition. *An ordered self-resolution inference*

$$\frac{\mathcal{D}(A)}{\mathcal{D}(\bot)\,,\mathcal{D}(\top)}$$

is redundant in N*, whenever some clause* $\Gamma\,,\mathcal{D}'(\top)\,,\mathcal{D}'(\bot)$ *logically follows from clauses in* N *smaller than* $\mathcal{D}$*, where* $\mathcal{D}$ *can be written as* $\Gamma, \mathcal{D}'(A)$ *with* A *occurring in* $\mathcal{D}'$ *and possibly also in* Γ*.*

By *simple resolution* we mean a variant of general ordered resolution where replacement in the main premise is confined to a single formula, as shown in Figure 8. The refutational completeness of simple resolution follows from the above propositions and Theorem 5.5. Note that the formula F can be chosen don't-care non-deterministically, as long as it contains at least one occurrence of a resolved atom. Once an inference has been computed for a specific formula F (or otherwise shown to be redundant), then any other inference with a different choice for F is also redundant.

5.4.4. Simplification

An important application of redundancy is in the use of logical equivalences for simplification of clauses. For instance, suppose $N, \mathcal{C}' \models \mathcal{C}$ and $N, \mathcal{C} \models \mathcal{C}'$ and $\mathcal{C} \succ \mathcal{C}'$. Then there is a two-step derivation,

$$N, \mathcal{C} \rhd N, \mathcal{C}, \mathcal{C}' \rhd N, \mathcal{C}'$$

where the first step is by deduction, as $\mathcal{C}'$ is a logical consequence of $N, \mathcal{C}$; and the second by deletion, as $\mathcal{C}$ is rendered redundant by $\mathcal{C}'$. We thus obtain a derived inference rule:

[5] These partial inferences are sound but need not satisfy the restrictions about selection.

Simple ordered resolution with selection

$$\frac{\mathcal{C}_1(A_1) \quad \ldots \quad \mathcal{C}_n(A_n) \quad \mathcal{D}\,, F(A_1, \ldots, A_n)}{\mathcal{C}_1(\bot)\,, \ldots, \mathcal{C}_n(\bot)\,, \mathcal{D}\,, F(\top, \ldots, \top)}$$

where F is a formula such that (i) either $A_1, \ldots, A_n$ is selected by S in $(\mathcal{D}, F)$, or else $S(\mathcal{D}, F)$ is empty, $n = 1$, and A_1 is maximal in $(\mathcal{D}, F)$, (ii) each atom A_i is maximal in $\mathcal{C}_i$, and (iii) no clause $\mathcal{C}_i$ contains a selected atom.

Simple ordered self-resolution

$$\frac{\mathcal{D}, F(A)}{\mathcal{D}, F(\bot), F(\top)}$$

where (i) the atom A is maximal in $\mathcal{C}$, and (ii) $\mathcal{D}, F$ contains no selected atom.

Figure 8: *Simple ordered resolution* SO

Simplification

$$N, \mathcal{C} \rhd N, \mathcal{C}'$$

if $N, \mathcal{C}' \models \mathcal{C}$ and $N, \mathcal{C} \models \mathcal{C}'$ and $\mathcal{C} \succ \mathcal{C}'$

For example,

$$N, (\mathcal{C}\,, \bot) \rhd N, \mathcal{C}$$

is a simplification step.

A more interesting case of simplification is the use of object-level equivalences $F \equiv G$ for rewriting. More specifically, we get

$$N, (F \equiv G), \mathcal{C}[F] \quad \rhd \quad N, (F \equiv G), \mathcal{C}[G] \qquad \text{if } \mathcal{C}[F] \succ \mathcal{C}[G]$$

Equivalences occur in many problem domains and very often simplification is the natural way of dealing with them. For example, an equivalence $X \subseteq (Y \cap Z) \equiv [(X \subseteq Y) \wedge (X \subseteq Z)]$ can be used to replace any occurrence of $X \subseteq (Y \cap Z)$ by a conjunction of simpler "subset relations" $X \subseteq Y$ and $X \subseteq Z$. We believe that simplification in this sense may also be the right context for an analysis of the question of "demodulation across argument and literal boundaries," a research problem posed by Wos [1988].

Meta-level equivalences suitable for simplification can be conveniently described by rewrite systems. For example, by P we denote the set of the following rewrite rules for elimination of $\bot$ and $\top$ from conjunctions, disjunctions and negations:

$$\begin{array}{rcl@{\qquad}rcl}
\alpha \wedge \bot & \Rightarrow & \bot & \bot \wedge \alpha & \Rightarrow & \bot \\
\alpha \wedge \top & \Rightarrow & \alpha & \top \wedge \alpha & \Rightarrow & \alpha \\
\alpha \vee \bot & \Rightarrow & \alpha & \bot \vee \alpha & \Rightarrow & \alpha \\
\alpha \vee \top & \Rightarrow & \top & \top \vee \alpha & \Rightarrow & \top \\
\neg\bot & \Rightarrow & \top & \neg\top & \Rightarrow & \bot
\end{array}$$

If $F \Rightarrow F'$ is a ground instance of a rule in P, then $F \simeq F'$. Furthermore, the rewrite system is contained in any simplification ordering (including lexicographic path orderings). Consequently,

$$N, \mathcal{C} \rhd N, \mathcal{C}' \qquad \text{if } \mathcal{C} \Rightarrow^+_{\mathsf{P}} \mathcal{C}'$$

is a simplification step (for any simplification ordering).

Similar rules for the elimination of $\top$ and $\bot$ can be designed for other connectives, e.g.,

$$\begin{array}{rcl@{\qquad}rcl}
\alpha \supset \bot & \Rightarrow & \neg\alpha & \bot \supset \alpha & \Rightarrow & \top \\
\alpha \supset \top & \Rightarrow & \top & \top \supset \alpha & \Rightarrow & \alpha \\
\alpha \equiv \bot & \Rightarrow & \neg\alpha & \bot \equiv \alpha & \Rightarrow & \neg\alpha \\
\alpha \equiv \top & \Rightarrow & \alpha & \top \equiv \alpha & \Rightarrow & \alpha
\end{array}$$

cover implication and equivalence. The simplification rules for eliminating $\top$ and $\bot$ are often directly built into specialized variants of inferences for specific classes (normal forms) of formulas as discussed below.

5.4.5. Normal Forms

Rewrite systems also provide a convenient way of describing various normal forms. Let us briefly discuss negation, conjunctive, and disjunctive normal form. First note that all connectives can be expressed in terms of disjunction, conjunction and negation, as expressed by the rules,

$$\begin{array}{rcl}
\alpha \equiv \beta & \Rightarrow & (\alpha \supset \beta) \wedge (\beta \supset \alpha) \\
\alpha \supset \beta & \Rightarrow & \neg\alpha \vee \beta
\end{array}$$

for the case of implication and equivalence. Termination of these rules can be proved by a lexicographic path ordering in which the symbols to be eliminated ($\equiv$ and $\supset$ in this case) have higher precedence than the other connectives (here $\wedge$, $\vee$ and $\neg$).

We may then push negation inside and eliminate double negations:

$$\begin{array}{rcl}
\neg(\alpha \vee \beta) & \Rightarrow & \neg\alpha \wedge \neg\beta \\
\neg(\alpha \wedge \beta) & \Rightarrow & \neg\alpha \vee \neg\beta \\
\neg\neg\alpha & \Rightarrow & \alpha
\end{array}$$

Termination of these rules requires a precedence in which $\neg \succ \vee$ and $\neg \succ \wedge$. The normal forms defined by these rules are also called *negation normal forms*.

From negation normal form we get to *conjunctive normal form* by applying distributivity rules:

$$\begin{array}{rcl}(\alpha \wedge \beta) \vee \gamma & \Rightarrow & (\alpha \vee \gamma) \wedge (\beta \vee \gamma) \\ \alpha \vee (\beta \wedge \gamma) & \Rightarrow & (\alpha \vee \beta) \wedge (\alpha \vee \gamma)\end{array}$$

By C we denote the set consisting of all of the above rules. A lexicographic path ordering, in which $\neg \succ \vee \succ \wedge \succ \top \succ \bot$ and other connectives have higher precedence than $\neg$, can be used to prove termination of C. We emphasize that we are interested in the existence of normal forms (or "weak normalization," which is ensured by termination), but not in their uniqueness. Indeed, it is well-known that conjunctive normal forms are not unique. Also, formulas in conjunctive normal form may contain certain "redundancies." For example,

$$((A \vee A) \wedge (B \vee A)) \wedge ((A \vee \neg B) \wedge (B \vee \neg B))$$

is a conjunctive normal form of $(A \wedge B) \vee (A \wedge \neg B)$, which could be further simplified to

$$(A \wedge (B \vee A)) \wedge (A \vee \neg B).$$

These additional simplifications will be part of the transformation to standard clauses discussed below.

Disjunctive normal forms are obtained by distributing conjunctions over disjunctions:

$$\begin{array}{rcl}(\alpha \vee \beta) \wedge \gamma & \Rightarrow & (\alpha \wedge \gamma) \vee (\beta \wedge \gamma) \\ \alpha \wedge (\beta \vee \gamma) & \Rightarrow & (\alpha \wedge \beta) \vee (\alpha \wedge \gamma)\end{array}$$

To prove termination, we only need to slightly modify the above lexicographic path ordering, so that $\wedge \succ \vee$, instead of $\vee \succ \wedge$.

Inference rules can usually be more efficiently implemented when they need to be applied to clauses in normal form only. Normalization of the clauses is therefore often integrated directly into the computation of inferences, which formally leads to modified inference rules.

A function $\mathcal{N}$ from clauses to finite sets of clauses is called a *normal-form functor* (with respect to $\succ$) if for each clause $\mathcal{C}$ we have that (i) $\mathcal{C}$ is equivalent to the conjunction of all clauses in $\mathcal{N}(\mathcal{C})$ and (ii) $\mathcal{C} \succeq \mathcal{D}$; and $\mathcal{N}(\mathcal{D}) = \{\mathcal{D}\}$, for each clause $\mathcal{D}$ in $\mathcal{N}(\mathcal{C})$. The clauses $\mathcal{D}$ in $\mathcal{N}(\mathcal{C})$ are also called the *$\mathcal{N}$-normal forms* of $\mathcal{C}$. If $\mathcal{N}(\mathcal{C}) = \{\mathcal{C}\}$ the clause $\mathcal{C}$ is said to be *in $\mathcal{N}$-normal form*.

If Γ is an inference system, the *composition* $\mathcal{N} \circ \Gamma$ of Γ with $\mathcal{N}$ is defined to be the set of all inferences

$$\frac{\mathcal{C}_1 \ldots \mathcal{C}_n}{\mathcal{D}}$$

where all premises $\mathcal{C}_1, \ldots, \mathcal{C}_n$ are in $\mathcal{N}$-normal form, $\mathcal{D}$ is a clause in $\mathcal{N}(\mathcal{C})$, and $\mathcal{C}$ is the conclusion of an inference

$$\frac{\mathcal{C}_1 \ldots \mathcal{C}_n}{\mathcal{C}}$$

in Γ.

5.12. Theorem. *Let Γ be an inference system and $\mathcal{N}$ be a normal-form functor with respect to $\succ$. If Γ has the reduction property with respect to $\succ$ then $\mathcal{N} \circ \Gamma$ has the reduction property with respect to $\succ$ on clauses in $\mathcal{N}$-normal form.*

Proof. Suppose Γ has the reduction property with respect to a model functor I and the ordering $\succ$. Let N be a set of clauses in $\mathcal{N}$-normal form and $\mathcal{C}$ be a minimal counterexample for I_N in N, which is not a contradiction. Since Γ has the reduction property there is an inference with main premise $\mathcal{C}$ and side premises $\mathcal{C}_1, \ldots, \mathcal{C}_n$ in N, the conclusion $\mathcal{C}'$ of which is a smaller counterexample for I_N than $\mathcal{C}$. Since $\mathcal{C}'$ is equivalent to the conjunction of all the clauses in $\mathcal{N}(\mathcal{C}')$, some clause $\mathcal{D}'$ in $\mathcal{N}(\mathcal{C}')$ must also be a counterexample for I_N. The clause $\mathcal{D}'$ is the conclusion of an inference in $\mathcal{N} \circ \Gamma$ and is a smaller counterexample than $\mathcal{C}$. □

This theorem implies that if Γ has the reduction property, then its composition with a normal-form functor $\mathcal{N}$ is refutationally complete for clauses in $\mathcal{N}$-normal form. Note that it would suffice to require that $\mathcal{C}' \succ \mathcal{D}$, for each clause $\mathcal{D}$ in $\mathcal{N}(\mathcal{C})$, whenever $\mathcal{C}$ is the conclusion of an inference in Γ with main premise $\mathcal{C}'$ in $\mathcal{N}$-normal form.

5.13. Corollary. *If $\mathcal{N}$ is a normal-form functor with respect to $\succ$ and Γ has the reduction property with respect to $\succ$, then a saturated (with respect to $\mathcal{N} \circ \Gamma$) set of clauses in $\mathcal{N}$-normal form is either consistent or else contains a contradiction.*

Observe that an inference in Γ from $\mathcal{N}$-normal forms is redundant if and only if one of the corresponding inferences in $\mathcal{N} \circ \Gamma$ with the same premises is redundant. Moreover a derivation

$$N, C \rhd^* N, C, \mathcal{N}(C) \rhd^* N, \mathcal{N}(C)$$

in which one replaces a clause $\mathcal{C}$ by its normal forms, is an instance of simplification.

For example, we may replace a formula by its conjunctive normal form,

$$N, (\mathcal{C}, F) \rhd N, (\mathcal{C}, F') \qquad \text{if } F \Rightarrow^!_{\mathsf{C}} F'$$

and then eliminate conjunctions by a sequence of two deduction steps followed by a deletion,

$$\begin{aligned} & N, (\mathcal{C}\,, F \wedge G) \\ & \quad \rhd N, (\mathcal{C}\,, F \wedge G), (\mathcal{C}\,, F) \\ & \quad \rhd N, (\mathcal{C}\,, F \wedge G), (\mathcal{C}\,, F), (\mathcal{C}\,, G) \\ & \quad \rhd N, (\mathcal{C}\,, F), (\mathcal{C}\,, G) \end{aligned}$$

Disjunctions can be eliminated in a similar way,

$$N, (\mathcal{C}\,, F \vee G) \rhd N, (\mathcal{C}\,, F\,, G),$$

(Binary) ordered resolution with selection

$$\frac{C \vee A \quad D \vee \neg A}{C \vee D}$$

where (i) A is a maximal atom in the side premise and the side premise contains no selected atoms and (ii) the atom A is either selected by S in $D \vee \neg A$, or else $D \vee \neg A$ contains no selected atoms at all and A is maximal and non-positive in D.

Figure 9: *Binary ordered resolution* R

so that any general clause can be reduced to an equivalent finite set of standard clauses. Finally, we may get rid of additional redundancies by applying the simplification rules,

$$\begin{aligned} N, (\mathcal{C}\,, A\,, \neg A) \quad &\rhd \quad N, \top \text{ and} \\ N, \top \quad &\rhd \quad N. \end{aligned}$$

6. Basic Resolution Strategies

Standard resolution can be obtained as a combination of general resolution with a normal-form functor. In this section we outline how several well-known variants of standard resolution can be derived by appropriate settings of the parameters of general resolution. The refutational completeness of these "composite" inference systems follows from Theorem 5.12.

6.1. Binary Resolution

In the theorem proving literature clauses are often represented as sets, rather than multisets, of literals, so that the representation of inferences can be simplified (though there is a trade-off, as the lifting process is more complicated). The transition from multisets to sets can be captured by a simplification rule,

$$N, (\mathcal{C}\,, L\,, L) \quad \rhd \quad N, (\mathcal{C}\,, L),$$

which removes duplicate occurrences of literals from a (standard) clause.

Figure 9 depicts a variant of resolution, denoted by $\mathsf{R}^{\succ}_{S}$, that is intended for standard variable-free clauses that represent sets. Self-resolution does not need to be applied to standard clauses, where atoms always have a specific polarity, whether positive or negative; cf. Proposition 5.8.

The following result can be obtained as corollary to Theorems 5.12 and 5.5:

6.1. THEOREM. *Let S be a selection function and $\succ$ be an admissible ordering. If a set of clauses N is saturated up to redundancy under $\mathsf{R}_S^{\succ}$, then N is unsatisfiable if and only if it contains a contradiction.*

We emphasize that the notion of saturation up to redundancy is flexible enough to cover "mixed" derivations, in which both general and standard resolution inferences appear.

The removal of duplicate literals from a clause can also be expressed as an inference rule:

Positive ordered factoring

$$\frac{C \vee A \vee A}{C \vee A}$$

where A is maximal in C and no atom in C is selected.

Factoring is needed for formulas with variables and therefore its inclusion at the ground level yields a closer correspondence between inferences at the ground and non-ground level. For certain applications a negative version of factoring is needed:

Negative ordered factoring

$$\frac{C \vee \neg A \vee \neg A}{C \vee \neg A}$$

where A is selected or C contains no selected atom and A is maximal in C.

6.2. Hyper-Resolution

Let now S be a selection function which selects one, and only one, negative atom from a clause (if there is a negative occurrence of an atom at all). The only standard clauses from which no atoms can be selected are positive clauses, which contain no negative literals. This kind of selection function results in the following inference rule on standard clauses, which is essentially a special case of general positive ordered resolution, except that the conclusion has been simplified.

Positive ordered resolution

$$\frac{C \vee A \quad D \vee \neg A}{C \vee D}$$

where (i) C is a positive clause with maximal atom A and (ii) the atom A is selected by S in $D \vee \neg A$.

The opposite of this selection function for standard clauses is a "maximal" selection function, where the sequence of all negative atoms in a clause is selected. This leads to the following inference rule:

Ordered resolution with maximal selection

$$\frac{C_1 \vee A_1 \quad \ldots \quad C_n \vee A_n \quad D_N \vee D_P}{C_1 \vee \ldots \vee C_n \vee D_P}$$

where (i) the clauses $C_1, \ldots, C_n$ and D_P are all positive, (ii) no clause C_i contains any of the atoms A_j (not even A_i), (iii) D_N is a negative clause containing all the literals $\neg A_1, \ldots, \neg A_n$, and only those literals, and (iv) each atom A_i is maximal in the clause C_i.

Condition (ii) essentially formulates the restriction suggested by Proposition 5.7, though we use the fact that multiple positive occurrences of maximal atoms can be eliminated by ordered positive factoring.

Resolution with maximal selection is closely related to hyper-resolution [Robinson 1965*a*]. If $C_1 \vee A_1, \ldots, C_n \vee A_n$ are positive (standard) clauses, where A_i is maximal in C_i, for all i; and there exist (standard) clauses $D_1, \ldots, D_{n+1}$, such that D_{i+1} is a resolvent between C_i and D_i on A_i, and D_{n+1} is positive, then

$$\frac{C_1 \vee A_1 \quad \ldots \quad C_n \vee A_n \quad D_0}{D_{n+1}}$$

is called a *(positive) hyper-resolution* inference. The first n premises are called the *electrons*, the last premise, the *nucleus* of the inference. Since all the premises are standard clauses, the final (positive) clause D_{n+1} (but not the intermediate clauses $D_1, \ldots, D_n$) is independent of the order in which the electrons are listed.

Now consider the premises of an inference by ordered resolution with maximal selection and denote by D_i the simplified version of

$$C_1 \vee \ldots \vee C_i \vee D_N[A_1/\top, \ldots, A_i/\top] \vee D_P,$$

for $i = 0, 1, \ldots, n$. Then D_{i+1} is the conclusion of a (standard) resolution inference on A_i with premises $C_i \vee A_i$ and D_i. In other words, each ordered resolution inference with maximal selection is a hyper-resolution inference. But there are hyper-resolution inferences, such as

$$\frac{A_1 \vee A_2 \quad A_2 \vee A_3 \quad \neg A_1 \vee \neg A_2 \vee A_4}{A_2 \vee A_3 \vee A_4}$$

that are not ordered resolution inferences with maximal selection. (The atom A_2 should not occur in the first premise.) In short, ordered resolution inference with maximal selection is a more restrictive inference system than hyper-resolution.

We should also point out that a resolution inference with maximal selection is redundant if any of the "intermediate resolvents" Γ_i is redundant (cf. Proposition 5.10). This answers an open question in [Wos 1988].

6.3. Boolean Ring-Based Methods

Let AC be the set of (two-way) rewrite rules

$$\begin{aligned}
(\alpha \wedge \beta) \wedge \gamma &\;\Leftrightarrow\; \alpha \wedge (\beta \wedge \gamma) \\
\alpha \wedge \beta &\;\Leftrightarrow\; \beta \wedge \alpha \\
(\alpha \oplus \beta) \oplus \gamma &\;\Leftrightarrow\; \alpha \oplus (\beta \oplus \gamma) \\
\alpha \oplus \beta &\;\Leftrightarrow\; \beta \oplus \alpha
\end{aligned}$$

and BR the set of rewrite rules

$$\begin{array}{rcl}
\neg\alpha & \Rightarrow & \alpha \oplus \top \\
\alpha \vee \beta & \Rightarrow & (\alpha \wedge \beta) \oplus (\alpha \oplus \beta) \\
\alpha \wedge \bot & \Rightarrow & \bot \\
\alpha \wedge \top & \Rightarrow & \alpha \\
\alpha \wedge \alpha & \Rightarrow & \alpha \\
\alpha \oplus \bot & \Rightarrow & \alpha \\
\alpha \oplus \alpha & \Rightarrow & \bot \\
(\alpha \oplus \beta) \wedge \gamma & \Rightarrow & (\alpha \wedge \gamma) \oplus (\beta \wedge \gamma)
\end{array}$$

All of these rules describe logical equivalences. Furthermore, the rewrite system BR/AC terminates and the corresponding normal forms, called *BR-normal forms*, are unique up to equivalence under AC [Hsiang 1985]. We denote by $\Phi(F)$ a BR-normal form of F.

Normal forms may also be represented as (possibly empty) *sums*

$$P_1 \oplus P_2 \oplus \cdots \oplus P_n$$

of (pairwise different, possibly empty) *products* of (pairwise different) atoms

$$P_i = A_{i,1} \ldots A_{i,n_i}$$

where a product $A_1 A_2 \ldots A_n$ represents the formula

$$(A_1 \wedge (A_2 \wedge \cdots (A_{n-1} \wedge A_n) \cdots),$$

a sum $P_1 \oplus P_2 \oplus \cdots \oplus P_n$ represents the formula

$$(P_1 \oplus (P_2 \oplus \cdots (P_{n-1} \oplus P_n) \cdots),$$

the empty product denotes $\top$, and the empty sum denotes $\bot$. We often need to single out a specific atom and write

$$AP_1 \oplus \ldots \oplus AP_m \oplus Q_1 \oplus \ldots \oplus Q_n$$

with the understanding that none of the products $P_1, \ldots, P_m, Q_1, \ldots, Q_n$ contains A. We often use $AP \oplus Q$ as a short form for such a sum of products.

We will now design a refutationally complete calculus for formulas in BR-normal form by composing general ordered resolution with the normal-form functor Φ for generating BR-normal forms. Consider a general ordered resolution inference

$$\frac{AP \oplus Q \quad AP' \oplus Q'}{(\bot \wedge P) \oplus Q\,, (\top \wedge P') \oplus Q'}$$

Ordered BR-resolution

$$\frac{AP \oplus Q \qquad AP' \oplus Q'}{\Phi((Q \oplus \top)(P' \oplus Q') \oplus Q)}$$

where both premises are BR-normal forms with maximal atom A.

BR-self-resolution

$$\frac{AP \oplus Q}{\Phi(PQ \oplus P \oplus Q)}$$

where the premise is a BR-normal form with maximal atom A.

Figure 10: *Ordered BR Resolution* $\mathsf{BR}^{\succ}$

---

in which both premises are sums of products in normal form. The disjunction

$$((\bot \wedge P) \oplus Q) \vee ((\top \wedge P') \oplus Q'),$$

which is logically equivalent to the resolvent, can be simplified to

$$Q \vee (P' \oplus Q').$$

Eliminating the disjunction symbol from the latter formula we get

$$Q(P' \oplus Q') \oplus Q \oplus (P' \oplus Q')$$

which is equivalent to

$$(Q \oplus \top)(P' \oplus Q') \oplus Q.$$

These considerations lead to the inference system $\mathsf{BR}^{\succ}$ displayed in Figure 10. BR is monotone with respect to resolution inferences on formulas in BR-normal form since the maximal atom A of the two premises does not appear in the conclusion and its BR-normal form. (For general clauses, however, we usually have $BR(\mathcal{C}) \succ \mathcal{C}$ if we treat the "," in $\mathcal{C}$ as disjunction $\vee$ when applying BR.) Therefore we may apply Theorem 5.12 and obtain as a consequence of Theorem 5.4:

6.2. Theorem. *Let N be a set of formulas in BR-normal form. If N is saturated with respect to* BR *and $\mathcal{R}^{\succ}$, then N is inconsistent if and only if it contains a contradiction.*

An alternative to ordered BR-resolution is a form of critical pair computation for equations between sums of products of atoms. A sum $AP \oplus Q$ with maximal atom A is viewed as an equation $AP \oplus Q \approx \top$ and oriented into a rewrite rule $AP \Rightarrow Q \oplus \top$. Given another rewrite rule $AP' \Rightarrow Q' \oplus \top$ with the same maximal atom, there is a critical pair $P(Q' \oplus \top) \approx P'(Q \oplus \top)$ (between the AC-extensions of the rules) that can itself be represented by the polynomial corresponding to $P(Q' \oplus \top) \oplus P'(Q \oplus \top) \oplus \top$. Hence (simple) BR-superposition is the following inference:

Simple BR-superposition

$$\frac{AP \oplus Q \qquad AP' \oplus Q'}{\Phi(P(Q' \oplus \top) \oplus P'(Q \oplus \top) \oplus \top)}$$

where both premises are BR-normal forms with maximal atom A.

This inference rule is sound. In addition, we have

$$A,\, AP \oplus Q \models P \equiv (Q \oplus \top)$$

and therefore also

$$\begin{array}{ll} A,\, AP \oplus Q \;\models & \\ & P(Q' \oplus \top) \oplus P'(Q \oplus \top) \oplus \top \\ & \equiv \quad (Q \oplus \top)(Q' \oplus \top) \oplus P'(Q \oplus \top) \oplus \top \\ & \equiv \quad (Q \oplus \top)(Q' \oplus \top \oplus P') \oplus \top \\ & \equiv \quad (Q \oplus \top)(P' \oplus Q') \oplus Q. \end{array}$$

The calculation shows that if A and $AP \oplus Q$ are true, the equivalence of the conclusions of a BR-superposition inference and a BR-resolution inference with the same premises follows. For proofs of redundancy of resolution inferences, the resolved atom and the positive premises may be assumed (cf. Proposition 5.6). Hence, the redundancy of a BR-superposition inference implies the redundancy of the corresponding ordered BR-resolution inference (with the same premises). Denoting by BRS the inference system of BR-superposition and BR-self-resolution, we therefore obtain the following completeness result:

6.3. THEOREM. *Let N be a set of formulas in BR-normal form. If N is saturated with respect to* BRS *and $\mathcal{R}^{\succ}$, then N is inconsistent if and only if it contains a contradiction.*

Let us next briefly describe how a positive variant of BR-resolution can be derived from general positive resolution. First note that a product $A_1 \dots A_n$ is false in the interpretation $I_\bot$ (in which all atoms are false), unless it is the trivial product $\top$. Consequently, a formula in BR normal form is false in $I_\bot$ if, and only if, it does not contain the trivial product $\top$.

Positive BR-resolution

$$\frac{AP \oplus Q \qquad AP' \oplus Q'}{\Phi((Q \oplus \top)(P' \oplus Q') \oplus Q)}$$

where both premises are BR-normal forms, the first premise contains no trivial product $\top$, and A is the maximal atom in the first premise.

On the other hand, a product $A_1 \dots A_n$ is always true in the interpretation $I_\top$ (in which all atoms are true). Thus, a formula in BR normal form is false in $I_\top$ if, and only if, it contains an even number of products and is different from $\bot$. The negative variant of ordered BR-resolution is therefore of the following form:

Negative BR-resolution

$$\frac{AP \oplus Q \qquad AP' \oplus Q'}{\Phi((Q \oplus \top)(P' \oplus Q') \oplus Q)}$$

where both premises are BR-normal forms, the first premise is a sum of an even number of products, and A is the maximal atom in the first premise.

Since positive BR-resolution (together with positive BR-self-resolution) is simply the composition of general positive resolution $\mathsf{GP}^{\succ}$ (which is itself an instance of general resolution $\mathsf{O}_S^{\succ}$ for a specific selection function S) with the normal form functor Φ for BR-normal forms, refutational completeness with respect to standard redundancy follows. The same is true, dually, for the negative variant of BR-resolution (together with negative BR-self-resolution) .

It is clear that variants of these inference rules similar to BR-superposition are also refutationally complete. Incidentally, the completeness of the latter inference systems was posed as an open problem by Zhang [1994], who introduced negative BR-resolution and also mentions its positive dual. In [Zhang 1994] the negative variant of BR-resolution is called the "odd strategy," which may seem odd, given that the inference is characterized by the syntactic restriction that the first premise consist of an *even* number of products. However, Zhang represents the formula $AP \oplus Q$ by an equation $\Phi(AP \oplus Q \oplus \top) \approx \bot$, so that a formula with an even number of products is turned into an equation, the left-hand side of which consists of an odd number of products.

A Boolean ring-based method for first-order theorem proving was first described by Hsiang [1985]. This so-called "N-strategy" is closely related to (standard) negative resolution.[6] The method applies to equations $A \approx \bot$ where A is a sum of products obtained from the negation of a given *clause*. That is, the initial formulas are assumed to be clauses, the negations of which are translated to sums of products. For instance, the negation of a negative clause $\neg P \vee \neg Q$ is represented by an equation $PQ \approx \bot$ (called an "N-rule") with a single product of atoms on the left-hand side. The N-strategy is a resolution method with the restriction that one of the premises of each inference be an N-rule. Thus, standard negative resolution is a special case. The N-strategy also allows for simplification by rewriting whereby equations may be transformed so that they no longer represent single standard clauses. However, the method is only complete if rather severe restrictions are imposed on simplification [Zhang 1994]. Thus, the N-strategy in essence more closely resembles standard negative resolution, whereas negative-BR-resolution is a true non-clausal method, as shown above.

There are also slightly different approaches that do not derive from non-clausal resolution, but where critical pair computations and other techniques from associative-commutative completion are directly applied to the rewrite system BR/AC and polynomial equations; see [Kapur and Narendran 1985] and [Bachmair and Dershowitz 1987] for details.

[6] Various improvements of the original N-strategy have been proposed, e.g., [Müller and Socher-Ambrosius 1988]

7. Refined Techniques for Defining Orderings and Selection Functions

The basic resolution strategies described in the preceding section can be further refined by more elaborate definitions of orderings and selection functions. Key techniques in this regard are renamings of formulas and conservative extensions of a theory. For instance, if a positive literal is renamed to become negative, it can then be selected, so that renaming indirectly allows for a selection of positive literals. A conservative extension of a theory by new symbols, on the other hand, provides more freedom in defining a clause ordering. The semantics of new symbols needs to be specified by suitable formulas. The first step in the theorem proving process then often consists of saturating the collection of all of these "definitions," whereas later steps correspond to refinements of resolution in the presence of such a pre-saturated subset.

7.1. Renaming and Semantic Resolution

There are several refinements of resolution that are essentially syntactic variants of inference systems described in the previous sections. In particular, this is the case for semantic resolution [Slagle 1967] and hence *set-of-support resolution*. Semantic resolution is defined with respect to a given Herbrand interpretation I:

Semantic resolution

$$\frac{C \vee L \quad D \vee \overline{L}}{C \vee D}$$

where (i) L and $\overline{L}$ are complementary literals, (ii) $C \vee L$ is false in I, and (iii) L is the maximal literal in the first premise.

A well-known refinement of resolution that is covered by semantic resolution relies on a *set of support* [Wos, Robinson and Carson 1965]. Let T (the "theory") be a satisfiable subset of a set of clauses N. A resolution inference

$$\frac{C \vee L \quad D \vee \overline{L}}{C \vee D}$$

obeys the set-of-support restriction for T if at least one premise is a clause in $N \setminus T$ (the "set of support").

Let us denote by $I_\perp$ the Herbrand interpretation in which all atoms are false, and by $I_\top$ the interpretation in which all atoms are true. Semantic resolution with respect to $I_\perp$ corresponds to positive resolution;[7] whereas semantic resolution with respect to $I_\top$ has been called *negative resolution*. Positive and negative resolution are dual to each other in that positive resolution is based on the minimal (in a set-theoretic sense) Herbrand interpretation $I_\perp$, whereas negative resolution is based on the maximal Herbrand interpretation $I_\top$. It turns out that semantic resolution

[7]For simplicity, we disregard selection, which has historically not been used in the context of semantics resolution.

with respect to an interpretation I may be seen as a syntactic variant of semantic resolution with respect to any other interpretation I'. We outline how semantic resolution may be mapped to positive resolution via renaming of literals, so that the previous completeness results are applicable.

Let $\mathcal{L}$ be the set of all atoms $A_1, A_2, \ldots$ in a given language and $\mathcal{L}'$ be a disjoint set with atoms $A'_1, A'_2, \ldots$. By a *renaming* we mean a mapping ρ from $\mathcal{L}$ to the set of literals over $\mathcal{L}'$. Most of the renamings ρ we will use satisfy the condition that no two literals $\rho(A_i)$ and $\rho(A_j)$ are complementary. Renamings can be homomorphically extended to clauses and sets of clauses, typically with the modification that a double negation $\neg\neg A$ is replaced by A.

If I' is a Herbrand interpretation over $\mathcal{L}'$ and ρ a renaming from $\mathcal{L}$ to $\mathcal{L}'$ then by I'_ρ we denote the Herbrand interpretation that consists of all atoms A in $\mathcal{L}$ for which $\rho(A)$ is in I'. Therefore, a formula $\rho(F)$ is true in I' if and only if F is true in I'_ρ.

If I is a Herbrand interpretation over $\mathcal{L}$, the *renaming ρ_I induced by I* is defined as follows: $\rho_I(A_i) = \neg A'_i$, if $A_i \in I$, and $\rho_I(A_i) = A'_i$, if $A_i \notin I$. In other words, ρ_I maps atoms that are false in I to positive literals, and atoms that are true in I to negative literals. For instance, $\rho_I(I_\perp) = \rho_I(\emptyset) = I$. A renamed clause $\rho_I(C)$ is true in $I_\perp$ if and only if C is true in I. Furthermore, if

$$\frac{C \vee L \quad D \vee \overline{L}}{C \vee D}$$

is a semantic resolution inference with respect to I, then

$$\frac{\rho_I(C \vee L) \quad \rho_I(D \vee \overline{L})}{\rho_I(C \vee D)}$$

is a positive resolution inference. In short, semantic resolution with respect to any interpretation I may be viewed as a syntactic variant of positive resolution.

7.1. Proposition. *Let N be a consistent set of standard clauses. Then there exist a renaming ρ and a selection function S such that $\rho(N)$ is saturated up to redundancy by ordered resolution $\mathsf{R}^{\succ}_S$ with respect to S and any ordering $\succ$.*

Proof. Let I be a model of N and ρ_I be the renaming induced by I. Observe that no clause in $\rho_I(N)$ is positive. Let S be any selection function that chooses at least one atom in each non-positive clause. Then each clause in $\rho_I(N)$ contains a selected atom, which implies that there is no inference by ordered resolution with this selection from $\rho_I(N)$. □

The significance of this proposition lies in the fact that it asserts the existence of a saturated presentation for any consistent theory, which also implies the refutational completeness of set-of-support resolution.

7.2. Resolution with Free Selection

Selection functions can be used to single out arbitrary negative occurrences of atoms in a clause. The observations in the preceding section show that positive occurrences can be selected indirectly via renaming, though additional clauses defining new symbols need to be introduced. In certain cases, selection can be applied freely to both positive and negative occurrences of atoms.

A *free selection function* is a mapping on clauses that selects exactly one (positive or negative) atom from each (nonempty) clause. By *binary resolution with free selection* we mean a resolution inference in which an atom is resolved only if it is selected in *both* premises. Completeness results for binary resolution with free selection for Horn clauses have been proved by Lynch [1997] and de Nivelle [1996]. We present a simple proof based on a specific kind of renaming.

A *Horn clause* is a standard clause with at most one positive literal. Let N be a set of Horn clause over a language $\mathcal{L}$ and $\mathcal{L}'$ be the set of all expressions $o(A,n)$, where A is an atom in $\mathcal{L}$, n is a natural number, and o is a new binary symbol. The renaming ρ from $\mathcal{L}'$ to $\mathcal{L}$ is defined by $\rho(o(A,N)) = A$.

7.2. Theorem. *If a set of Horn clauses N is saturated under binary resolution resolution with free selection, then N is either consistent or else contains the empty clause.*

Proof. Let N be a set of clauses over $\mathcal{L}$. We say that a set of clauses N' over $\mathcal{L}'$ encodes N if

(i) $\rho(C')$ is a clause in N, for each clause C' in N',

(ii) if C is a clause $\neg A_1 \vee \cdots \vee \neg A_k$ in N and $n_1, \ldots, n_k$ are natural numbers, then $\neg o(A_1, n_1) \vee \cdots \neg o(A_k, n_k)$ is a clause in N', and

(iii) if C is a clause $\neg A_1 \vee \cdots \vee \neg A_k \vee B$ in N and $n_1, \ldots, n_k$ are natural numbers, then there is a clause C' in N' of the form $\neg o(A_1, n_1) \vee \cdots \neg o(A_k, n_k) \vee o(B, n)$, where $n > \sum_{i=1}^{k} n_i$.

It can easily be shown that N' is satisfiable whenever N is satisfiable. The converse is also true. For if M is the set of all resolvents from N, then the set $N' \cup K$ encodes $N \cup M$, where K is the set of all resolvents from N'. Therefore, if the empty clause can be derived from N, it can also be derived from N'.

Let now $\succ$ be any admissible ordering on clauses over $\mathcal{L}'$, such that $o(A,n) \succ o(B,m)$ whenever $n > m$. In addition, if S is a free selection function for N, we define a regular selection function S' for N', such that S' selects the same negative atom from a clause as S, but selects no atom when S selects a positive atom. There is a correspondence between binary resolution inferences with S from N and ordered resolution inferences with selection function S' from N', as positive atoms are maximal in the ordering $\succ$. The refutational completeness of ordered resolution $\mathsf{R}_S^{\succ}$ thus implies the completeness of resolution with free selection, for Horn clauses. □

A consequence of this theorem is the completeness of SLD-resolution [Kowalski

1974], as used in Prolog, where in each program clause the positive atom (the "head") is selected and in each negative (or "goal") clause some negative atom is selected.

Resolution with free selection (together with unrestricted positive and negative factoring) is generally incomplete for non-Horn clauses. For example, consider the inconsistent set of clauses,

$$A \vee \underline{B},\ \underline{A} \vee \neg B,\ \neg\underline{A} \vee B,\ \neg A \vee \neg\underline{B},$$

where selection is indicated by underlining. By resolution we can derive only tautologies, $B \vee \neg\underline{B}$ and $A \vee \neg\underline{A}$, respectively. Even if these tautologies are not eliminated, with a selection as indicated no new clauses can be derived. But de Nivelle [1996] has shown that free selection for full clauses may cause incompleteness only when every resolution-based refutation requires at least one step of factoring. That is the case in particular for every inconsistent set of binary clauses, as resolution between two binary clauses produces again a binary clause and shorter clauses can only be obtained by factoring.

SL-resolution [Kowalski and Kuehner 1971] is a resolution strategy for full clauses that also employs a rather liberal selection strategy, but we do not know how to justify this strategy within the semantic framework of reducing counterexamples. More specifically, SL-resolution is a refinement of set-of-support resolution where arbitrary, positive or negative atoms may be selected, provided they have been introduced by a theory clause premise of a previous resolution step. It is closely related to model elimination [Loveland 1969] and semantic tableaux. In Sections 8.2 and 8.3 we shall briefly describe how to generalize the linear theorem proving derivations of Section 4.1 to derivation trees. This will allow us to model semantic tableaux and refinements such as model elimination in our framework. We shall in particular see that aspects of selection that cannot be modeled on the semantic level of partial interpretations can often be justified on the level of derivations.

7.3. Conservative Extensions

A straightforward transformation of a formula to clausal form or conjunctive normal form—for instance, via normalization with the rewrite system C—may exponentially increase the size of a formula. Fortunately, there are transformation schemes that preserve the consistency of a formula, but avoid an exponential increase in size. They are based on extending the given language by new predicate symbols and corresponding definitions.

Let N be a set of formulas, F be an expression that occurs as a subformula in N. and L be a literal P or $\neg P$, where the atom P does not appear in any formula in N. We say that a set N', M is a *regular extension* of N (by L) if (i) N' is obtained from N by replacing one or more occurrences of F by L and (ii) M is logically equivalent to $L \equiv F$. More specifically, we speak of a *positive* (respectively, *negative*) *extension* if only positive (respectively, negative) occurrences of the subformula F are replaced and M is logically equivalent to $L \supset F$ (resp., $F \supset L$). Finally, we say that a set

K is a *(conservative) extension* of N if it is obtained from N by a sequence of one or more (regular, positive, and/or negative) extensions.

7.3. Proposition. *If K is an extension of N, then K is consistent if, and only if, N is consistent.*

Proof. Let I be a model of $N[F]$ and L be a literal P or $\neg P$, where P is not contained in N. If L and F have the same truth value in I, then I is a model of any (regular, positive, or negative) extension of $N[F]$ by L. If L and F have different truth values in I, define I' to be the interpretation $I \cup \{P\}$, if $P \notin I$, and $I \setminus \{P\}$, if $P \in I$. Then L and F have the same truth value in I' (since P does not occur in F), and I' is a model of any (regular, positive, or negative) extension of $N[F]$ by L. This implies that consistency is preserved by any conservative extension.

For the other direction, suppose first that $K = N', M$ is a regular extension of N by L and I is a model of a K. Then L and F have the same truth value in I and, hence, N' and $N[F]$ also have the same truth value in I. Since I is a model of N', it has to be a model of $N[F]$ as well.

If I is a model of a positive extension N', M of $N[F]$ by L, then it is a model of N' and of $L \supset F$. Again, if L and F have the same truth value in I, then $N[F]$ is true in I. If L and F have different truth values in I, then L must be false, and F true, in I. Since N' results from replacing positive occurrences of F in N, for any such occurrence $C[F]$ within a clause C of N the clause $C[F/\top]$ is a tautology, which implies that $C[F]$ is true in I. Clauses in N in which F is not replaced also occur in N' and, hence, are true in I by assumption. Thus, I is a model of N.

The case of negative extensions is handled in a similar way. In sum, this implies that if a regular, positive or negative extension of N is consistent, so is N itself. □

We illustrate the use of the extension principle by showing how a formula can be converted to an equivalent set of standard clauses so that the size increases only by a constant factor, cf., Tseitin [1970].

Let $E_P(L \wedge L')$ be a set of three standard clauses,

$$\neg P \vee L, \neg P \vee L', P \vee \overline{L} \vee \overline{L'},$$

where L and $\overline{L}$, and also L' and $\overline{L'}$, are complementary pairs of literals. Similarly, let $E_P(L \vee L')$ be the set

$$\neg P \vee L \vee L', P \vee \overline{L}, P \vee \overline{L'};$$

$E_P(L \supset L')$ the set

$$\neg P \vee \overline{L} \vee L', P \vee L, P \vee \overline{L'};$$

and $E_P(L \equiv L')$ the set

$$\neg P \vee \overline{L} \vee L', \neg P \vee L \vee \overline{L'}, P \vee L \vee L', P \vee \overline{L} \vee \overline{L'}.$$

We have the following logical equivalences:

$$\begin{aligned} E_P(L \wedge L') &\simeq P \equiv (L \wedge L') \\ E_P(L \vee L') &\simeq P \equiv (L \vee L') \\ E_P(L \supset L') &\simeq P \equiv (L \supset L') \\ E_P(L \equiv L') &\simeq P \equiv (L \equiv L') \end{aligned}$$

Let now N be a finite set of clauses, where we assume for simplicity that all formulas are in negation normal form. (The transformation to negation normal form increases the size of a formula by a constant factor only.) If N is not a set of standard clauses, it must contain a subformula $L \circ L'$, where $\circ$ is one of the connectives $\wedge$, $\vee$, $\supset$ or $\equiv$ and L and L' are literals. Then $N[P_{L\circ L'}], E_{P_{L\circ L'}}(L \circ L')$ is an extension of $N[L \circ L']$, where $P_{L\circ L'}$ is a new predicate symbol. Each extension step eliminates at least one occurrence of a binary connective, so that we eventually end up with a set of standard clauses that is consistent if, and only if, the initial set N is consistent. In the worst case each occurrence of a logical connective in the initial formula has to be replaced by a new atom and at most four additional clauses, each with no more than three literals. Thus, the size of the initial set N may increase only by a constant factor.

Plaisted and Greenbaum [1986] have presented a refinement of this transformation scheme in which the polarities of abbreviated formulas $L \circ L'$ are considered so that for a positive [negative] $L \circ L'$ a positive [negative] extension by $P_{L\circ L'}$ is generated. They also discuss how to automatically extend an ordering to the new predicate symbols in such a way that symbols that represent small formulas are preferred in ordered inferences.

7.4. Lock Resolution

Extension results in interesting variations of resolution, such as lock resolution [Boyer 1971], which can essentially be encoded by positive hyper-resolution.

Lock resolution is applied to standard clauses in which each occurrence of a literal has been assigned a positive integer, called a *lock index*. For example, in the following set N_0 of four clauses,

$$\begin{array}{lcr} {}_8A & \vee & {}_7B, \\ {}_6\neg A & \vee & {}_5\neg B, \\ {}_4B & \vee & {}_3\neg A, \\ {}_2\neg B & \vee & {}_1A \end{array}$$

each literal occurrence has been assigned a unique index, but in general different literal occurrences may be assigned the same index. The lock restriction states that only literals with a maximal index must be resolved.[8] More formally, we have the

[8]We have departed from the original definition in [Boyer 1971], which restricts resolution to minimal literals, so as to avoid confusion with ordered resolution, which resolves maximal literals.

Lock resolution

$$\frac{C \vee {}_iA \quad {}_j\neg A \vee D}{C \vee D}$$

where no literal in C has a larger index than i, and no literal in D has a larger index than j.

Figure 11: *Lock resolution*

---

inference rule given in Figure 11. For example,

$$\frac{{}_8A \vee {}_7B \qquad {}_6\neg A \vee {}_5\neg B}{{}_7B \vee {}_5\neg B}$$

is a lock resolution inference, but

$$\frac{{}_8A \vee {}_7B \qquad {}_4B \vee {}_3\neg A}{{}_7B \vee {}_4B}$$

is not.

Let $N = C_1, \ldots, C_n$ be a finite set of standard clauses (with lock indices). For each clause

$$C_i = {}_{l_{i,1}}A_{i,1} \vee \cdots \vee {}_{l_{i,k_i}}A_{i,k_i} \vee {}_{l_{i,k_i+1}}\neg B_{i,k_i+1} \vee \cdots \vee {}_{l_{i,k_i+m_i}}\neg B_{i,k_i+m_i}$$

let C_i' be the (renamed) clause

$$C_i = P_{i,1} \vee \cdots \vee P_{i,k_i} \vee P_{i,k_i+1} \vee \cdots \vee P_{i,k_i+m_i}$$

where the $P_{i,j}$ are new predicate constants not occurring in N, and let M_i be the set of all clauses $\neg P_{i,j} \vee A_{i,j}$, where $1 \leq j \leq k_i$, and $\neg P_{i,k_i+l} \vee \neg B_{i,k_i+l}$, where $1 \leq l \leq m_i$. We also say that $P_{i,j}$ *encodes* the corresponding literal, $A_{i,j}$ or $\neg B_{i,j}$, respectively. We assume that any two of the $P_{i,j}$ are identical if, and only if, the encoded literals are identical and their associated lock indices are the same. Finally, let N' be the (renamed) set $C_1', \ldots, C_n'$ and M be the set $M_1, \ldots, M_n$. The clauses in M are called *definitions*.

Note that a clause $\neg P_{i,j} \vee A_{i,j}$ is logically equivalent to the implication $P_{i,j} \supset A_{i,j}$, and $\neg P_{i,j} \vee \neg B_{i,j}$ is logically equivalent to $P_{i,j} \supset \neg B_{i,j}$. Thus, N', M is an extension of N.

For example, from the matrix N_0 above we get a renamed matrix

$$\begin{array}{ccc} P_{1,1} & \vee & P_{1,2} \\ P_{2,1} & \vee & P_{2,2} \\ P_{3,1} & \vee & P_{3,2} \\ P_{4,1} & \vee & P_{4,2} \end{array}$$

where the predicate constants $P_{i,j}$ are defined by this matrix M:

$$\begin{array}{lllllll} \neg P_{1,1} & \vee & A & \qquad & \neg P_{1,2} & \vee & B \\ \neg P_{2,1} & \vee & \neg A & & \neg P_{2,2} & \vee & \neg B \\ \neg P_{3,1} & \vee & B & & \neg P_{3,2} & \vee & \neg A \\ \neg P_{4,1} & \vee & \neg B & & \neg P_{4,2} & \vee & A \end{array}$$

Let now $\succ$ be an ordering in which (i) $A_{i,j} \succ P_{i,j}$ and $B_{i,j} \succ P_{i,j}$, for all i, j, and (ii) $P_{i,j} \succ P_{i',j'}$ if, and only if, the lock index $l_{i,j}$ (associated with the literal encoded by $P_{i,j}$) is greater than the lock index $l_{i',j'}$ (associated with the literal encoded by $P_{i',j'}$). We then saturate the set M under ordered resolution. This results, with the given ordering, in the elimination of the "old" atoms $A_{i,j}$, that is, the result is a set M, K, where K consists of all two-element negative clauses $\neg P_{i,j} \vee \neg P_{k,l}$, such that $P_{i,j}$ and $P_{k,l}$ encode complementary literals. The clauses in K are called *connections*.

For the above example we obtain the following connections:

$$\begin{array}{lllllll} \neg P_{1,1} & \vee & \neg P_{2,1} & \qquad & \neg P_{1,2} & \vee & \neg P_{2,2} \\ \neg P_{1,1} & \vee & \neg P_{3,2} & & \neg P_{1,2} & \vee & \neg P_{4,1} \\ \neg P_{3,1} & \vee & \neg P_{4,1} & & \neg P_{4,2} & \vee & \neg P_{2,1} \\ \neg P_{3,1} & \vee & \neg P_{2,2} & & \neg P_{4,2} & \vee & \neg P_{3,2} \end{array}$$

For instance, the connection $\neg P_{1,1} \vee \neg P_{3,2}$ indicates that in the original matrix N_0 the first literal in the first clause is complementary to the second literal in the third clause.

The set N', M, K is partially saturated in the sense that all inferences with premises from M (i.e., definitions) are redundant in this context. If we use a selection function that selects both literals in a connection, then any possible ordered resolution inference with this selection must be of the form

$$\frac{C \vee P_{i,j} \quad D \vee P_{k,l} \quad \neg P_{i,j} \vee \neg P_{k,l}}{C \vee D}$$

where $P_{i,j}$ is strictly maximal in the first positive premise and $P_{k,l}$ is strictly maximal in the second positive premise. The negative premise is a connection and the conclusion is again a positive clause. In other words, these are positive hyper-resolution inferences with connections as nucleus. The ordering restrictions guarantee that these hyper-resolution inferences encode lock resolution inferences. More precisely, if $C' \vee L$ denotes the clause obtained from $C \vee P_{i,j}$ by replacing each atom by the literal it encodes, and $D' \vee \overline{L}$ is obtained in the same way from $D \vee P_{k,l}$, then

$$\frac{C' \vee L \quad D' \vee \overline{L}}{C' \vee D'}$$

is a lock resolution inference. Conversely, each lock resolution inference is encoded by a positive hyper-resolution inference of the above form. In sum, there is a one-to-one correspondence between hyper-resolution inferences (with two renamed clauses as negative premises and a connection as positive premise) and lock resolution (on the original clauses).

For example, the hyper-resolution inference

$$\frac{P_{3,1} \vee P_{3,2} \quad P_{4,1} \vee P_{4,2} \quad \neg P_{3,1} \vee \neg P_{4,1}}{P_{3,2} \vee P_{4,2}}$$

encodes the lock resolution inference

$$\frac{{}_4B \vee {}_3\neg A \qquad {}_2\neg B \vee {}_1A}{{}_3\neg A \vee {}_1A}$$

We should emphasize that lock resolution, as described above, needs to be combined with factoring (either as inference or as simplification in derivations). The following "syntactic" version of factoring is sufficient:

$$\frac{C, {}_iL, {}_iL}{C, {}_iL}$$

where L is a literal, and no literal in C has a greater index than i. The correspondence between inferences on original and encoded formulas is not one-to-one in this case. Positive, ordered factoring on encoded formulas correspond to positive or negative lock factoring inferences on the original clauses. The correspondence becomes one-to-one if the same predicate symbol is used to encode different occurrences of an indexed literal.

Boyer [Boyer 1971] describes lock resolution in terms of "semantic factoring," where a literal ${}_iL$ is deleted from a clause that contains a literal ${}_jL$ with $j > i$, but also mentions the syntactic variant. Semantic factoring is preferable for ground clauses, whereas syntactic factoring is probably a better choice for clauses with variables.

Boyer also mentions that lock resolution is not compatible with tautology deletion. For example, the two lock resolution inferences we have shown at the beginning of this section are the only ones from premises in N_0. In each case, the conclusion is a tautology. If these inferences were regarded as redundant, then N_0 would be saturated up to redundancy. The calculus would thus be incomplete, as N_0 is inconsistent and contains no contradiction. This observation appears to contradict the fact that our completeness results cover redundancy. But redundancy in our sense has to be applied to encoded clauses and inferences and we can see from the example that the encoding, $P_{3,2} \vee P_{4,2}$, of the tautology ${}_3\neg A \vee {}_1A$ is not itself a tautology and therefore is not redundant.

In some applications of lock resolution as a decision procedure for certain decidable fragments of first-order logic one needs to be able to decrease lock indices don't-care non-deterministically. We will show how to justify such steps within the framework of standard redundancy. A *lock index reduction clause* is a clause of the form $\neg Q \vee P$, where P and Q are "lock symbols" that encode the same original literal L and $P \succ Q$. (We may assume that for each literal L over the given language and lock index i we have a lock symbol Q encoding the indexed literal ${}_iL$, together with the defining clause for Q.) Lock index reduction clauses encode tautologies. They may be viewed as implications $Q \supset P$, to be applied backwards to decrease lock indices. As the positive literal is maximal, the only possible inferences with a lock reduction clause are of the form

$$\frac{\neg Q \vee P \quad \neg P \vee \neg P'}{\neg Q \vee \neg P'}$$

where a connection in K is the main premise and another connection is the conclusion. (As these partial hyper-inferences are, therefore, redundant we need not consider any "complete" hyper-inferences from $\neg Q \vee \neg P'$ in which one of the side premises is a renamed clause in N'.) It is sound to add these connections initially and no additional resolution inferences result therefrom. The transition

Lock index reduction

$$N,\ C \vee {}_iL \ \rhd\ N,\ C \vee {}_iL,\ C \vee {}_jL \ \rhd\ N,\ C \vee {}_jL$$

is admissible in theorem proving derivations based on lock resolution, if $i > j$ and if C contains a literal ${}_kL'$ such that $k > i$. (The encoding of $C \vee {}_jL$ is consistency-preserving. Moreover, if the encoding of N contains the lock index reduction clause corresponding to the encodings of the indicated occurrences of ${}_jL$ and ${}_iL$, respectively, the encoding of $C \vee {}_iL$ becomes redundant, provided the lock index reduction clause is be smaller than the encoding of $C \vee {}_iL$, which is the case if C contains a literal ${}_kL'$ with $k > i$.)

The condition, that C contain a suitable literal ${}_kL'$, can be dispensed with when lock index reduction takes place immediately after a clause has been derived in an inference. In the following theorem, the "redundancy" of an inference on original clauses refers to the standard redundancy of the corresponding hyper-resolution inference on the encoded clauses.

7.4. Theorem. *Let*

$$\frac{C \vee {}_iA \quad {}_j\neg A \vee D}{C \vee D}$$

be an inference by lock resolution. The inference is redundant whenever the inference

$$\frac{C \vee {}_iA \quad {}_j\neg A \vee D}{C' \vee D'}$$

is redundant, where C' and D' result from C and D, respectively, by decreasing the lock indices of some of the literals.

Proof. The lock indices in $C \vee D$ are smaller than or equal to the maximum of i and j. Therefore the lock index reduction clauses that are needed to justify the encoding of the implication $(C' \vee D') \supset (C \vee D)$ are smaller than the main premise in the encoding of the inference (which is a connection between ${}_iA$ and ${}_j\neg A$). $\square$

A related result has been obtained by de Nivelle [1996], who also applied it to derive completeness proofs for ordered resolution based on certain so-called "non-liftable" orderings.

In sum, lock resolution is essentially ordered resolution, where initial *occurrences* of atoms (denoted by the labels) are ordered. Accordingly, selection is free in initial clauses. However, since labels are inherited, selection is not free on derived clauses.

7.5. The Inverse Method

The inverse method was proposed by Maslov [1964]. Its basic inference rules are formulated in terms of a given set of (generalized) disjunctions of conjunctions of formulas. In our description of the method we follow Lifschitz [1989], where the method is formulated for disjunctions $G_1 \vee \ldots \vee G_k$ of conjunctions $G_i = L_{i,1} \wedge \ldots \wedge L_{i,m_i}$ of literals $L_{i,j}$. The disjunctions have been called "super-clauses", and the conjunctions G_i, "super-literals" in [Lifschitz 1989]. The negation $\neg G_i$ of a super-literal, which is equivalent to a standard clause, is denoted by $\overline{G}_i$. Given a set of input super-clauses, an *S-clause* is any standard clause that is logically equivalent to a disjunction of negated super-literals $\overline{G}_i$ in the input. For example, given the input

$$\begin{array}{l}(P \wedge \neg Q) \vee (R \wedge T)\\ \neg P \vee Q\\ \neg R\end{array}$$

the input super-literals are the conjunctions

$$P \wedge \neg Q,\ R \wedge T,\ \neg P,\ Q,\ \neg R,$$

and their negations are the clauses

$$\neg P \vee Q,\ \neg R \vee \neg T,\ P,\ \neg Q,\ R\ .$$

Forming, for instance, the disjunction consisting of the negated first and third super-literals yields the clause

$$\neg P \vee Q \vee P$$

which is a tautology. They represent connections between complementary literals in the input super-clauses.

The inverse method consists of the following two inference rules:

Type A inferences

$$\frac{}{C}$$

where C is any S-clause which is a tautology.

Type B inferences

$$\frac{E_1 \vee \overline{G}_1 \quad \ldots \quad E_k \vee \overline{G}_k}{E_1 \vee \ldots \vee E_k}$$

where $G_1 \vee \ldots \vee G_k$ is an input super-clause, and where the premises are S-clauses.

Clearly, the conclusion of a type B inference is again an S-clause. In [Lifschitz 1989] factoring is built into the set notation for clauses.

We will show how to encode this standard version of the inverse method by positive hyper-resolution. The encoding will be similar to the encoding of lock resolution. Let N be a set of formulas $F_1, \ldots, F_n$, where each F_i is a disjunction $G_{i,1} \vee \ldots \vee G_{i,m_i}$ of conjunctions $G_{i,j}$ of literals. For each conjunction $G_{i,j}$ we introduce a propositional constant $P_{i,j}$ that does not occur in N, and denote by $M_{i,j}$ a standard clause logically equivalent to $\neg P_{i,j} \supset G_{i,j}$. By M we denote the set of all clauses $M_{1,1}, \ldots, M_{n,m_n}$. Let C_i be the clause $\neg P_{i,1} \vee \ldots \vee \neg P_{i,m_i}$ and N' be the set of standard clauses $C_1, \ldots, C_n$. Then N', M is an extension of N. (Note that it is sufficient to introduce one constant $P_{i,j}$ for all occurrences of a formula $G_{i,j}$; we need not introduce different constants for different occurrences of $G_{i,j}$ in N.) In the example N' has the clauses

$$\neg P_{1,1} \quad \vee \quad \neg P_{1,2} \qquad (1)$$
$$\neg P_{2,1} \quad \vee \quad \neg P_{2,2} \qquad (2)$$
$$\neg P_{3,1} \qquad (3)$$

with implications

$$\begin{array}{rcl} \neg P_{1,1} & \supset & P \wedge \neg Q \\ \neg P_{1,2} & \supset & R \wedge T \\ \neg P_{2,1} & \supset & \neg P \\ \neg P_{2,2} & \supset & Q \\ \neg P_{3,1} & \supset & \neg R \end{array}$$

and, hence, a set M of clauses,

$$\begin{array}{rcr} P_{1,1} & \vee & P \\ P_{1,1} & \vee & \neg Q \\ P_{1,2} & \vee & R \\ P_{1,2} & \vee & T \\ P_{2,1} & \vee & \neg P \\ P_{2,2} & \vee & Q \\ P_{3,1} & \vee & \neg R \end{array} \quad .$$

As can be observed from the example, in $\neg P_{i,j} \supset G_{i,j}$ (the "definition" of the literal $\neg P_i, j)$, if $G_{i,j}$ is a conjunction $L^1_{i,j} \wedge \cdots \wedge L^{m_{i,j}}_{i,j}$, then the implication is logically equivalent to the set of binary clauses $P_{i,j} \vee L^1_{i,j}, \ldots, P_{i,j} \vee L^{m_{i,j}}_{i,j}$. Let $\succ$ be an admissible ordering in which all new atoms $P_{i,j}$ are smaller than all old atoms occurring in N. If we saturate M under ordered resolution, the result is a set M, K, where K consists of positive clauses of the form $P_{i,j} \vee P_{i',j'}$. The clauses in K encode the tautologies that can be obtained by "Type A" inferences. In the example, K consists of the clauses

$$
\begin{array}{lclr}
P_{1,1} & \vee & P_{2,1} & (4) \\
P_{1,1} & \vee & P_{2,2} & (5) \\
P_{1,2} & \vee & P_{3,1} & (6).
\end{array}
$$

Let us also use a selection function that selects all negative literals in a clause. Then the non-redundant resolution inferences during saturation of N', M, K are positive hyper-resolution inferences of the form

$$\frac{D_1 \vee L_1 \quad \ldots \quad D_n \vee L_n \quad \neg L_1 \vee \cdots \vee \neg L_n}{D_1 \vee \cdots \vee D_n}$$

where the positive premises are positive clauses (initially from K) and the negative premise is from N'. The conclusion is again a positive clause. These inferences correspond to "Type B" inferences (The negative premise $\neg L_1 \vee \cdots \vee \neg L_n$ encodes one of the formulas in the original set N.) Conversely, any "Type B" inference can be translated into a positive hyper-resolution inference of this form. In short, we have established a one-to-one correspondence between the (standard version of the) inverse method and positive hyper-resolution. For refutational completeness, ordered factoring for positive clauses has to be added.

In the example one derives a contradiction by the following series of type B inferences:

$$
\begin{array}{llr}
(7) & P_{1,2} & [\ (6) \text{ into } (3)\ ] \\
(8) & P_{2,1} & [\ (4) \text{ and } (7) \text{ into } (1)\ ] \\
(9) & P_{1,1} & [\ (5) \text{ and } (8) \text{ into } (2)\ ] \\
(10) & \bot & [\ (7) \text{ and } (9) \text{ into } (1)\ ]
\end{array}
$$

Maslov's super-clauses represent a particular (non-standard) clausal normal form. Specializing simple ordered resolution SO to super-clauses would result in an inference

$$\frac{\mathcal{C}, A \wedge G \quad \mathcal{D}, \neg A \wedge H}{\mathcal{C}[A/\bot], \mathcal{D}}$$

which is related to (a sequence of two) type B inferences of the inverse method but with slight differences in the way multiple occurrences of the resolved atom A are handled.

7.6. Ordered Theory Resolution

Theory resolution, a concept introduced by Stickel [1985], refers to resolution inferences that have been specially designed for a given consistent set of clauses T, called the *theory*. (Clauses not in T are also called *goal clauses*.) A minimal requirement for a theory resolution calculus is that explicit inferences within the theory be excluded, so that T needs to be saturated in an appropriate manner. In Section 7.1 we have shown that by renaming one can always obtain a saturated presentation for a consistent set of standard clauses. In fact, set-of-support resolution may be viewed as an instance of theory resolution, though the presentation of T underlying set-of-support resolution is trivially saturated: renaming renders theory clauses non-positive and selection of the non-positive parts therefore permits only inferences in which at least one premise is a goal clause. No non-trivial consequences of T are explicitly represented. More powerful instances of theory resolution can be obtained if the theory T is saturated in a nontrivial way.

Let us first illustrate some of the technical issues. Consider a theory consisting of a single clause specifying that p is a transitive predicate. Note that every (non-ground) binary resolution inference with the transitivity clause $p(x, y) \wedge p(y, z) \supset p(x, z)$ will introduce an extra variable in the resolvent that is not present in the other premise of the inference. Furthermore, unification is no effective filter for inferences as any arbitrary atom $p(s, t)$ can be unified, say, with $p(x, y)$. These issues can be addressed by hyper-inferences in which two literals of the transitivity clause are resolved simultaneously. Then no new variables are introduced and the corresponding unification problem is non-trivial due to the fact that any two literals of the transitivity clause share a common variable. The strategy, admitted by the general theory of ordered resolution, of selecting the two negative literals avoids inferences between theory clauses and generates no extra variables, but is not very useful in practice as it leads to an enumeration of the entire transitive closure of p and, hence, is not goal-oriented enough. A better approach is to select the two literals that contain the maximal of the three (instantiated) *terms* x, y, and z (with respect to a given well-founded ordering on ground terms). The resulting inferences would, on the ground level, eliminate the maximal term from any "reachability" problem involving the transitive predicate p:[9]

Ordered chaining, positive

$$\frac{C \vee p(s,t) \quad D \vee p(t,u) \quad p(s,t) \wedge p(t,u) \supset p(s,u)}{C \vee D \vee p(s,u)}$$

where (i) $t \succ s$, $t \succ u$, (ii) $p(s, t)$ is strictly maximal with respect to C, and (iii) $p(t, u)$ is strictly maximal with respect to D

Ordered chaining, negative (I)

$$\frac{C \vee p(s,t) \quad D \vee \neg p(s,u) \quad p(s,t) \wedge p(t,u) \supset p(s,u)}{C \vee D \vee \neg p(t,u)}$$

where (i) $s \succ t$, $s \succ u$, (ii) $p(s, t)$ is strictly maximal with respect to C, and (iii) $p(s, u)$ is maximal with respect to D.

[9] We present the ground versions of inference systems, but will use clauses with variables in examples.

Ordered chaining, negative (II)

$$\frac{C \vee p(t,u) \quad D \vee \neg p(s,u) \quad p(s,t) \wedge p(t,u) \supset p(s,u)}{C \vee D \vee \neg p(s,t)}$$

where (i) $u \succ t$, $u \succeq s$, (ii) $p(t,u)$ is strictly maximal with respect to C, and (iii) $p(s,u)$ is maximal with respect to D.

This inference system, together with binary ordered resolution and factoring for non-transitivity clauses, turns out to be refutationally complete (for suitable orderings), but involves the selection of non-maximal positive literals in the transitivity clause. The concept of ordered theory resolution which we describe next is centered around more flexible selection strategies for theory clauses. It comes as no surprise that this, again, implicitly involves specific forms of renamings. These inferences will be based on (total) term orderings that have to be extended in an appropriate way to (usually only partial) orderings on atomic formulas.

Let $\succ$ be a *partial*, well-founded ordering on (ground) atoms and T be a set of (ground) clauses. We will consider a calculus in which all maximal atoms in a theory clause from T are resolved simultaneously by hyper-inferences and where inferences between theory clauses are redundant. In the case of transitivity, a suitable ordering can be defined in terms of a well-founded, total ordering on ground terms: $p(s,t) \succ p(s',t')$ if, and only if, either (i) $\max(s,t) \succ \max(s',t')$ (in the term ordering) or (ii) $\max(s,t) = \max(s',t')$ and $s = t$, but $s' \neq t'$. Note that two atoms are uncomparable if they have the same maximal term as one argument and different, non-maximal terms as the other argument.

The description of theory resolution will be facilitated by writing literals as *signed atoms* $+A$ and $-A$, where a *sign* is either "+" or "−". The signed atom $+A$ denotes the positive literal A, while $-A$ denotes the negative literal $\neg A$. The two signs are considered to be complements of each other. We use the letters σ and τ to denote signs and denote by $\overline{\sigma}$ the complement of σ.

By ordered theory resolution with respect to a theory T, an ordering $\succ$ on atoms, and a total and well-founded extension $\succ'$ of $\succ$, we mean the following inference rule.

Ordered theory resolution

$$\frac{C_1 \vee \sigma_1 A_1 \quad \ldots \quad C_k \vee \sigma_k A_k}{C_1 \vee \cdots \vee C_k \vee D}$$

where no premise is in T, but T contains a clause $\overline{\sigma}_1 A_1 \vee \ldots \vee \overline{\sigma}_k A_k \vee D$ such that (i) $B \succeq' A_i$ for no atom B in C_i, (ii) the atoms A_i are pairwise incomparable with respect to $\succ$, and (iii) for each atom B in D there exists an i such that $A_i \succ B$.

In essence, an ordered theory resolution corresponds to a hyper-resolution of goal clauses into a theory clause that resolves all the maximal (with respect to $\succ$) atoms of the theory clause simultaneously. A (smaller) residuum of non-maximal atoms from the theory clause forms part of the conclusion of the inference. Only maximal atoms in goal clauses are resolved, where maximality is determined by the admissible extension $\succ'$ of the partial ordering $\succ$.

The combination of (i) ordered theory resolution and (ii) the restriction of ordered resolution and (positive and negative) ordered factoring to goal clauses, is refutationally complete for presentations that contain sufficiently many logical consequences of the given theory. A key to the completeness proof is to employ a signature extension and renaming so that (i) the maximal atoms in theory clauses become negative literals and, hence, selectable and (ii) goal clauses become positive. This leads to positive hyper-resolution inferences with goal clauses as electrons (and conclusions) and theory clauses as nucleus.

For each propositional symbol A in T we introduce two new symbols, denoted by A_+ and A_-, respectively.[10] If $\succ'$ is any extension of $\succ$, we order these new atoms as follows: $A_\sigma \succ' B_\tau$ if $A \succ' B$. The intended semantics of the new atoms is captured by *positive and negative connections*, $A_+ \vee A_-$ and $\neg A_+ \vee \neg A_-$, respectively. By K, we denote the set of all connections; by K_- the set of all negative connections; and by K_+ the set of all positive connections.

If C is a clause in T of the form

$$C = \sigma_1 A^1 \vee \ldots \vee \sigma_k A^k \vee \tau_1 B^1 \vee \ldots \vee \tau_m B^m$$

where the atoms A_i are pairwise incomparable under $\succ$ and each atom B^j is smaller than some atom A^i, then by $\rho(C)$ we denote the renamed clause

$$\neg A^1_{\overline{\sigma_1}} \vee \ldots \vee \neg A^k_{\overline{\sigma_k}} \vee B^1_{\tau_1} \vee \ldots \vee B^m_{\tau_m} \ .$$

For example, if C is the clause $\neg A \vee \neg B \vee C$ with maximal atoms B and C, then $\rho(C) = \neg B_+ \vee \neg C_- \vee A_-$.

Note that a clause C_0 logically follows from clauses $C_1, \ldots, C_k$ if, and only if, $\rho(C_0)$ logically follows from $\rho(C_1), \ldots, \rho(C_k)$ and K.

we call T a *saturated theory* if for all clauses $C \vee A \vee \ldots \vee A$ and $D \vee \neg A \vee \ldots \vee \neg A$ in T such that A is maximal (in $\succ$) with respect to C and D, but does not occur in C or D, the clause $\rho(C \vee D)$ logically follows from $\rho(T) \cup K_- \cup K_+^{\prec A}$, where $K_+^{\prec A}$ is the set of positive connections $B_+ \vee B_-$ for which $A \succ B$. (Note that according to this definition inferences from premises in which a maximal atom occurs both positively and negatively need not be considered.)

At first sight it may seem strange that arbitrary clauses in $\rho(T)$ are admitted in the above definition. But since ρ renders all maximal literals negative, little can be inferred from renamed theory clauses in the absence of positive connections. In particular, it is impossible to infer $\rho(C \vee D)$ directly from the renamed premises $\rho(C \vee A \vee \ldots \vee A) = \neg A_- \vee \ldots \vee \neg A_- \vee C'$ and $\rho(D \vee \neg A \vee \ldots \vee \neg A) = \neg A_+ \vee \ldots \neg A_+ \vee D'$ without the connection $A_+ \vee A_-$. A key condition of the above definition is that the positive connections are restricted to atoms smaller than the resolved atom A.

[10]The reader should not confuse the meta-level concept of signed atoms σA with the object-level symbols A_+ and A_-. The former simply served as a means to present theory resolution in a concise manner while the latter will be used to rename the original propositions.

If T consists of all ground instances of the transitivity clause for a predicate symbol p (over any given first-order signature) and the given ordering on atoms is based on a term ordering, then T is a saturated theory in this sense. For instance, consider the inference

$$\frac{p(s,t) \wedge p(t,u) \supset p(s,u) \quad p(s,u) \wedge p(u,v) \supset p(s,v)}{p(s,t) \wedge p(t,u) \wedge p(u,v) \supset p(s,v)}$$

where s is a maximal term and $p(s,u)$ a maximal atom in both premises. If s is a strictly maximal term, then renaming the conclusion yields the clause

$$\neg p(s,t)_+ \vee \neg p(s,v)_- \vee p(t,u)_- \vee p(u,v)_-.$$

The set $\rho(T)$ contains two other instances of transitivity:

$$\neg p(s,t)_+ \vee \neg p(s,v)_- \vee p(t,v)_-$$

and

$$\rho(p(t,u) \wedge p(u,v) \supset p(t,v)).$$

The maximal term in the latter clause is smaller than s. Therefore the clause contains only atoms smaller than $p(s,u)$ and hence all connections for its atoms are present in $K_- \cup K_+^{\prec p(s,u)}$. Thus $\rho(T) \cup K_- \cup K_+^{\prec p(s,u)}$ entails $\neg p(s,t)_+ \vee \neg p(s,v)_- \vee p(t,u)_- \vee p(u,v)_-$ which was to be shown. Suppose s is not strictly maximal. If $s = t$ or $s = u$, then one of the premises contains its maximal atom both negatively and positively, so that the inference need not be considered. If $s = v$, then we also have $s = u$ (for otherwise $p(s,u)$ would not be maximal) and the previous case applies.

7.5. Lemma. *Let T be a saturated theory. If I is a model of $\rho(T) \cup K_-$ then there exists an interpretation I' such that $I \subseteq I'$ and I' is a model of $\rho(T) \cup K$.*

Proof. Let $\succ'$ be any admissible total extension of $\succ$. We use induction on $\succ'$ to define, for all atoms A in T, interpretations I'_A and E_A as follows.

$$I'_A = I \cup \bigcup_{A \succ B} E_B.$$

If either A_+ or A_- is in I, then E_A is the empty set. Otherwise, E_A is the set $\{A_-\}$, if $I'_A \cup \{A_-\}$ is a model of $\rho(T)$, or else E_A is the set $\{A_+\}$. Finally,

$$I' = I \cup \bigcup_A E_A.$$

By construction I' is a model of K.

Suppose I' is not a model of $\rho(T)$. Let A be a minimal atom such that $I'_A \cup E_A$ is not a model of $\rho(T)$. Then I'_A is a model of $\rho(T) \cup K_- \cup K_+^{\prec A}$ and $E_A = \{A_+\}$. If it is not possible to extend I'_A by either A_+ or A_-, there exists a clause D' of the form $\neg A_+ \vee \ldots \vee \neg A_+ \vee D_1$ in $\rho(T)$ such that D_1 is false in I'_A, and there also exists a clause C' of the form $\neg A_- \vee \ldots \vee \neg A_- \vee C_1$ in $\rho(T)$ such that C_1 is false in I'_A. Moreover we may assume that neither C_1 nor D_1 contain $\neg A_+$ or $\neg A_-$. Suppose that $C \vee A$ and $D \vee \neg A$ are the corresponding clauses in T for which $\rho(C \vee A) = C'$ and $\rho(D \vee \neg A) = D'$, respectively, and consider a resolution inference on A from these. As the theory is saturated, $\rho(C \vee D)$ is entailed by $\rho(T) \cup K_- \cup K_+^{\prec A}$. As I'_A is a model for the latter set of clauses, it also satisfies $\rho(C \vee D)$. The two clauses $C_1 \vee D_1$ and $\rho(C \vee D)$ can differ only modulo renaming of literals. More precisely, a literal $+B_\sigma$ occurring in $C_1 \vee D_1$ might occur as $-B_{\overline{\sigma}}$ in $\rho(C \vee D)$. But this is possible only when $A \succ B$. Therefore, $K_- \cup K_+^{\prec A} \models (\rho(C \vee D) \supset (C_1 \vee D_1))$. In conclusion, $C_1 \vee D_1$ must be true in I'_A, which is a contradiction. □

Let us sketch how the completeness proof for ordered theory resolution can be completed. Consider the ordered resolution calculus $\mathsf{R}^{\succ'}$ with a selection function that selects all negative literals in $\rho(T)$ and K_-, and where $\succ'$ is a well-founded extension of $\succ$ that can itself be extended to a total admissible ordering. The given goal clauses are renamed into purely positive clauses, employing the new symbols. Positive connections are not considered for inferences. Then the only non-redundant inferences, apart from factoring are ordered hyper-resolution inferences with goal clauses as electrons and a nucleus that is either a renamed theory clause or one of the connections in K_-. The former inferences correspond to theory resolution, while the latter represent ordered resolution between two goal clauses. If no contradiction can be derived, the set of the renamed goal clauses plus $\rho(T)$ and K_- is satisfiable. If I is a model of this set, it can be extended by the above lemma to a model of K and $\rho(T)$. The renamed goal clauses are positive, and inferences create positive clauses only, so that the extension of the model also satisfies the latter.

A suitable notion of redundancy for theory resolution is represented by standard redundancy for ordered resolution with selection on the renamed goal and theory clauses, together with the connections K_-. In the presence of renaming, certain clauses which denote tautologies before the transformation need not stay redundant, a similar effect to what we have observed in lock resolution.

If T is saturated with respect to a total ordering $\succ$, there is no difference between an ordered theory resolution inference and an ordered resolution inference in which one premise is a goal clause and the second premise is a theory clause. A partial ordering may produce shorter residuums in theory resolution inferences. On the other hand one would expect that there are more theories that can be effectively saturated under a total atom ordering. Also note that atoms in non-theory predicates do not have to be renamed at all. Their negative occurrences in the original goal clauses can be freely selected.

Technically, the results in this section strictly extend both the results by Baumgartner [1992] and Bronsard and Reddy [1992]. The motivation of the first paper is the specialization of resolution to theories, whereas the second paper considers questions of decidability for saturated theories. We have seen in the example of a transitive relation that saturation with respect to a partial ordering may help to avoid the introduction of new variables in resolution inferences, which in turn is usually a requirement for decidability. In Bachmair and Ganzinger [1994] we have employed specific methods from term rewriting to obtain a calculus for dealing with transitive relations that is closely related to the chaining calculus that we have presented above as an instance of ordered theory resolution.

Other related approaches of building domain knowledge into resolution include resolution modulo an equational theory E, where syntactic unification is replaced by E-unification [Plotkin 1972], and constraint resolution [Huet 1972, Bürckert 1990]. In these approaches the setup is hierarchic and semantic in that the theory is represented by the set of its models and inconsistency means the failure of being able to extend any of these models to a model of the goal clauses. Bürckert's [1990] constraint resolution can be viewed as form of ordered theory resolution whenever the theory can be represented by a set of first-order clauses.

8. Global Theorem Proving Methods

8.1. The Davis-Putnam Method

A very effective method for testing the satisfiability of a finite set of variable-free standard clauses is the *Davis-Putnam method* [Davis and Putnam 1960]. This method combines simplification (unit reduction, pure literal detection) with a case analysis on atomic formulas. The original method described by Davis and Putnam can be modeled by self-resolution, combined with normalization to conjunctive form and various simplification techniques. But most implementations are based on a modification of the original method [Davis, Logemann and Loveland 1962]. This modified procedure can be formalized in terms of derivation trees, rather than the single "linear" theorem proving derivations we have used so far. The following rule may be viewed as a "tree expansion" rule:

Splitting

$N \;\rhd\; N, M_1 \quad | \quad \ldots \quad | \quad N, M_k \qquad (k \geq 1)$

where N is satisfiable if and only if one of the N, M_i is satisfiable

The application of splitting to a state N creates k new branches as indicated by the states N, M_i. That is, instead of a single derivation sequence we obtain a possibly infinite, but finitely branching tree, each node of which is labeled by a set of clauses.

Each branch in a derivation tree represents a theorem proving derivation, the limit of which needs to be saturated. More formally, if $N_0, N_1, N_2, \ldots$ is a branch in the tree, then its limit N_∞ is defined, as before, as the set $\bigcup_i \bigcap_{j \geq i} N_j$. The initial set N_0, labeling the root of the tree, is unsatisfiable if and only the limit of each branch is unsatisfiable. If the construction of a derivation tree is fair in the sense that the limit of each branch is saturated up to redundancy, then the set $\bigcup_j N_j$ of all formulas along a branch must contain a contradiction, whenever the limit set N_∞ is unsatisfiable. If a branch contains a contradiction, it need not be expanded further, but can be "closed." There are thus two possibilities: either all branches in a derivation tree can be closed, in which case the initial set N_0 is unsatisfiable; or else some branch can not be closed, in which case a model for N_0 can be defined via the limit N_∞ of the branch. We should point out that the inference system used for deductive inferences along a branch, and the redundancy criterion that controls deletion steps need not be uniform for all branches. For instance, different orderings may be used to restrict resolution steps along different branches. The key requirement is that each branch represent a fair theorem proving derivation (with respect to a refutationally complete inference system and an associated redundancy criterion).

The following rules, which describe the Davis-Putnam method, are based on specific instances of deduction, deletion and splitting. By $\overline{L}$ we denote the complement $\neg L$ of a literal L, with $\neg\neg A$ simplified to A.

Unit reduction

$N, L, C \vee \overline{L} \;\rhd\; N, L, C$

Unit subsumption

$N, L, C \vee L \;\rhd\; N, L$

Pure literal extension

$N \;\rhd\; N, L$ if L, but not $\overline{L}$, occurs in N, but not as a unit clause

Tautology deletion

$N, C \;\rhd\; N$ if C is a tautology

Splitting

$N \;\rhd\; N, A \;\mid\; N, \neg A$ if the atom A occurs in N

These rules are applied in a specific order. Splitting, in particular, is applied only if no other rule is applicable. Pure literal extension may trigger subsequent subsumption steps in which all clauses containing the literal L are eliminated.

It is easy to see that the Davis-Putnam method represents a fair strategy for sets of standard ground clauses. If no rule is applicable to a set of ground clauses N, then N either contains the empty clause or else is a set of literals, no two of which are complementary. In the first case, the set N is inconsistent; in the latter case, no resolution and/or factoring inferences are applicable, so that the set is saturated with respect to standard resolution $\mathsf{R}_S^{\succ}$ for any ordering $\succ$ and hence is consistent. The completeness of the method is, therefore, a consequence of Theorem 6.1. The Davis-Putnam method also terminates, provided the initial set of clauses is finite.

The method can be further improved by simplification and deletion steps such as general subsumption or subsumption resolution (see Section 4.3):

Subsumption resolution

$N, D \vee L, D \vee C \vee \overline{L} \;\rhd\; N, D \vee L, D \vee C$

Note that splitting on an atom A, followed by unit reduction with A and $\neg A$ on the two respective branches, is an instance of ordered self-resolution:

Splitting as self-resolution

$$\frac{M, F[A]}{M, F[A/\bot], F[A/\top]}$$

One may therefore adopt a different view and represent a derivation tree by a general clause. More specifically, each formula F in a clause M, F is itself in conjunctive normal form and represents a leaf of a partially expanded derivation tree. A self-resolution inference on an atom A in F represents splitting and unit reduction, as mentioned above. The ordering on atoms is defined "on the fly" in that the atom A that is resolved can be declared as the maximal remaining atom in F. The replacement of A by $\bot$ and $\top$ represents unit reduction. If A occurs with only one polarity in F, the conclusion can be simplified to either $M, F[A/\bot]$ or $M, F[A/\top]$.

It does not seem to be possible to model the above methods as a *linear* theorem proving process based on ordered resolution with selection in a natural way. The order in which atoms are resolved does not depend on a single uniform ordering, but is guided by syntactic characteristics of the involved subformulas. Typically, different orderings will be used for different formulas in a clause (which represent different branches in a derivation tree). But linear theorem proving derivations have to adopt a uniform ordering. (The method originally introduced by Davis and Putnam does employ a single ordering, though.)

On the other hand one can show that in a theorem proving derivation one may, without affecting the general results about how to effectively achieve fairness and, hence, the saturation of its limit, always admit *finitely* many "heureka" steps $N \rhd N'$ in which one replaces the set N by any set N' that preserves consistency or inconsistency. These observations indicate why the Davis-Putnam procedure does not easily extend to the infinite case of clauses with variables.

8.2. Saturated Semantic Tableaux

In the Davis-Putnam method splitting implicitly depends on tautologies of the form $A \vee \neg A$. *Semantic tableau methods* perform case splits directly on given clauses. For standard variable-free clauses we get two kinds of derivation steps:

Splitting on clauses

$$N, (L_1 \vee \ldots \vee L_k) \ \rhd \ N, L_1 \quad | \quad \ldots \quad | \quad N, L_k$$

Ancestor literal complement

$$N, L, \overline{L} \ \rhd \ \bot$$

Splitting on clauses is formally a combination of splitting and subsumption, where the original clause is eliminated via subsumption after the case split. Tableaux are specific, "regular" derivation trees in that no branch contains two nodes to which the same clause split has been applied. The application of the ancestor literal complement rule closes a branch. Formally, it combines a unit resolution step that generates the empty clause, with subsumption steps by which all other clauses are eliminated. If none of the above rules is applicable to a set N, then N consists either of the empty clause or else of unit clauses no two of which are complementary. Any strategy that applies the two rules exhaustively (and don't-care nondeterministically) is fair, and the limit of each branch in the derivation tree is closed under resolution. Refutational completeness again follows from Theorem 6.1.

The method can be improved by adding ordering constraints, selection functions and simplification, and exploiting the observation that it is sufficient to saturate the limit of each branch with respect to any (refutationally complete) variant of resolution. To that end we may restrict splitting on clauses so that it correspond to ordered resolution with selection.

Splitting on clauses induced by ordered resolution

$N, C \vee A \vee \ldots \vee A, D \vee \neg A \;\triangleright\; N, C, \neg A \quad | \quad N, A, D$

if $C \vee D$ is an ordered resolvent of $C \vee A \vee \ldots \vee A$ and $D \vee \neg A$ with respect to a given ordering and selection function.

In this formulation one does not split a clause into its k individual literals, but implicitly uses the tautology $A \vee \neg A$ to do a split into two branches that allows one to eliminate a clause in each branch. This approach is closely related to the Davis-Putnam method. In practice one difference between the two methods has been in the choice of concrete data structures to represent derivation trees. Tableaux are usually represented in a graphical way so that in an expansion

$$N, L_1 \vee \ldots \vee L_k \;\triangleright\; N, L_1 \quad | \quad \ldots \quad | \quad N, L_k$$

the set N that is common to all nodes is not duplicated, but implicitly shared. Even the initial clause set N_0 can be left implicit, so that only the literals L_i on which case splits are performed are stored explicitly. But then properties such as regularity have to be posed as extra constraints.

When one translates the above splitting rule with ordering restrictions and selection into the more common graphical formulations one obtains in particular a justification of the ordering restrictions for tableaux that have been proved complete by Klingenbeck and Hähnle [1994]. Again, it is clear that the ordering $\succ$ does not need to be uniform for all branches.

In short, the theoretical machinery of this paper suggests a natural generalization of the notion of a closed tableau to a *saturated tableau*, in which all branches are saturated up to redundancy with respect to one of the refutationally complete resolution calculi.

8.3. Model Elimination

Let us next consider sets of standard clauses of the form T, G, where G is a positive clause $A_1 \vee \ldots \vee A_k$ and each clause in T contains at least one negative literal. In fact, every inconsistent set of clauses contains such a subset (up to renaming). If N is inconsistent, then there exists an inconsistent finite subset T', G', where G' is a single clause and T' is consistent. We obtain a set T, G of the above form by renaming, as suggested by our discussion of semantic and set-of-support resolution in Section 7.1.

The set T is called the *theory* and G is called the *goal clause*. Let S be a selection function that selects exactly one negative literal in each clause in T. *Model elimination* [Loveland 1969] can be described by the following derivation rules:

Splitting on the goal clause

$$T, G \;\triangleright\; T, A_1 \quad | \quad \ldots \quad | \quad T, A_k$$

if $k > 1$

Expansion with a theory clause

$$N, (\neg A \vee L_1 \vee \ldots \vee L_n), A \;\triangleright\; N, A, L_1 \quad | \quad \ldots \quad | \quad N, A, L_n$$

if $\neg A$ is selected in the indicated (theory) clause and $n \geq 1$

Ancestor literal complement

$$N, L, \overline{L} \;\triangleright\; \bot$$

The expansion rule is a special case of the splitting rule on clauses from Section 8.2, where the branch for $\neg A$ has been omitted, as one can immediately derive $\bot$ from A and $\neg A$ and close the branch. The clause that is split is deleted to avoid a repeated case analysis for the same clause. Note that only theory clauses can be split, as all derived clauses consist of one literal only.

The key refinement in model elimination, as compared to the above semantic tableau method, is that the expansion with a theory clause must be triggered by a literal that is complementary to the selected literal in the clause. The literals that may trigger an expansion can all be traced back to the initial goal clause. In a fully expanded model elimination tree each branch is saturated up to redundancy under resolution with selection. Therefore, an initial clause T, G is inconsistent if and only if each branch in the tree ends with $\bot$.

The above rules have been formulated with respect to an initial clause T, G, which is a renaming of a subset T', G' of a given clause set. In practice one has to compute with the original clauses T', G', as a suitable renaming, and thus the selection of literals in theory clauses, is usually not known. In fact, even the right choice of the clause G' from $N \setminus T'$ may have to be guessed. In general, all different possible derivations have to be computed in a non-deterministic fashion.

Let us now return to the earlier question of free selection (cf., Section 7.2) and analyze the correspondence between model elimination trees and resolution derivations. In a model elimination tree, each node except the root is labeled either by the constant $\bot$ or else by a set of clauses that is obtained from the parent node by replacing a non-unit clause by a literal, which we call the *main literal* of such a node. Now take any derivation tree in which at least one expansion step was applied and where none of the leaf nodes contains a pair of complementary literals. Let $L_1, \ldots, L_m$ be the main literals of those leaves (from left-to-right) which are different from $\bot$. Then the clause $L_1 \vee \ldots \vee L_m$ can be derived from T, G by linear resolution (where at most one premise in each inference is a clause in T) plus (implicit or explicit) factoring. This observation provides an explanation for the linearity constraints in SL-resolution [Kowalski and Kuehner 1971]. In SL-resolution one may also select an arbitrary literal that was inherited by a resolvent from the theory clause premise of the resolution step. This simply corresponds, in the context of model elimination, to selecting the leaf where a derivation tree is to be expanded next. Obviously, one may don't-care non-deterministically choose any leaf. The ancestor literal complement steps in model elimination either correspond to implicit factoring or to subsumption resolution steps called "ancestor resolution" in [Kowalski and Kuehner 1971]. Tautology elimination and other redundancy elimination techniques described in the latter paper can be easily modeled by standard redundancy.

In conclusion, there is a close correspondence between model elimination trees and certain resolution derivations with simplification. Resolution methods, such as SL-resolution, that exploit this relationship may combine restrictions on the clause level, that can be justified semantically, with restrictions on the level of derivation trees, that can be justified by proof transformations.

9. First-Order Resolution Methods

9.1. First-Order Sequents

We have described resolution for (possibly infinite) sets of variable-free formulas. But resolution methods for propositional logic play only a minor role in practice compared to, say, the Davis-Putnam procedure (cf. Section 8.1) or methods based on ordered binary decision diagrams [Bryant 1992]. One of the main applications of resolution and other saturation-based methods is in automated theorem proving for first-order logic. Before resolution can be applied to a first-order formula, quantifiers have to be eliminated and formulas usually have to be converted to clause form. Sophisticated methods of clausal normal form transformation are described in [Baaz, Egly and Leitsch 2001, Nonnengart and Weidenbach 2001] (Chapters 5 and 6 of this Handbook). Inference rules and redundancy criteria need to be lifted to (quantifier-free) clauses with variables.

9.2. Lifting of Ordering Constraints

The key parameters in our description of theorem proving methods are orderings, selection functions, renaming strategies, and simplification and deletion techniques. Once these parameters are taken into account, lifting tends to become less straightforward.

In the literature we find two main techniques for lifting ordering constraints. One possibility is to define a "safe" approximation of ground constraints by extending a (total, well-founded) ground ordering to an (partial, well-founded) ordering on non-ground expressions, such that $E \succ E'$ if and only if $E\sigma \succ E'\sigma$, for all ground substitutions σ. If an inference is intended to capture ground instances for which, say, $E\sigma \succ E'\sigma$, one may add a constraint $E' \not\succeq E$ at the non-ground level. For instance (simple, binary) ordered resolution is approximated on the non-ground level by an inference rule (see also Figure 4)

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

where σ is a most general unifier of A and B and (i) $A'\sigma \not\succeq A\sigma$, for any A' in C, (ii) C contains no selected literal, and (iii) either $\neg B$ is selected or else $B'\sigma \not\succ B\sigma$, for any atom B' in D. It is better to apply the constraints to expressions to which the unifier has been applied as this may render a more precise approximation of the ordering constraints for the relevant ground instances of the inference. The satisfiability of non-ground constraints is undecidable for many orderings, so that one may have to resort to incomplete, but sound, constraint solvers (which represent a weaker approximation of ground constraints). In Section 4.3 we have proved a lifting lemma for binary ordered resolution with selection for this sort of approximation.

9.3. Constrained Formulas

A second, perhaps more adequate method is based on adding constraints to non-ground expressions, at the object level. A *constraint* restricts the set of ground terms that may substituted for variables. Notations such as

$$C \mathbin{[\![} \gamma$$

are used to denote the set of all those ground instances of a non-ground clause C for which the constraint γ is satisfied. For example, standard (binary) ordered resolution (without selection) can be lifted to an inference

$$\frac{C \vee A \mathbin{[\![} \gamma \quad D \vee \neg B \mathbin{[\![} \delta}{C \vee D \mathbin{[\![} \gamma \wedge \delta \wedge (A > C) \wedge (B \geq D) \wedge (A \approx B)}$$

on constrained non-ground clauses. The resolvent is constrained by the maximality conditions $(A > C) \wedge (B \geq D)$ and the equality constraint $A \approx B$, and also inherits the constraints γ and δ from the premises. The notation has been inspired by constraint logic programming and, in the context of automated theorem proving, formalized by Kirchner, Kirchner and Rusinowitch [1990] and others. Completeness proofs for certain saturation-based theorem proving strategies involving constrained clauses were first obtained by Huet [1972], Bürckert [1990] and Nieuwenhuis and Rubio [1992]. In moving the constraints from the meta-level to the object level, and using constraint inheritance, the ordering restrictions of the ground level are represented in a precise way and in principle no information is lost. Constraints are also significant when finite representations of saturated theories are sought, cf. Section 10.2.

These constraints combine logical and meta-logical restrictions in one expression. For example, in the above resolution inference, $A \approx B$ is the logical constraint that ensures the soundness of the inference, whereas the ordering constraints characterize meta-logical restrictions on inferences. There is some evidence that formalisms that explicitly separate logical from meta-logical constraints might be more appropriate. A meta-logical constraint can be relaxed without affecting the soundness and completeness of the inference system, and certain simplification strategies may actually require such constraint relaxation. Also, solvers for meta-logical constraints need not be complete. Logical constraints, on the other hand, must not be relaxed and the completeness of the theorem proving process requires that their solvability be decidable. Nieuwenhuis and Rubio [2001] (Chapter 7 of this Handbook) present equational theorem proving methods with constrained formulas in detail.

9.4. Resolution Modulo an Equational Theory

Constraints formalisms often provide a suitable framework for building *equational theories* (on ground atoms) into an inference system. One then considers only formulas constructed from the unique canonical representatives of term equivalence classes, a presentation that is preserved under resolution. For ordered inferences, one needs an ordering that is compatible with the equational theory and well-founded on canonical expressions. Refutational completeness requires that solvability of equality constraints be decidable. In cases where sets of unifiers may be very large (e.g., in AC-unification) or even infinite (e.g., in the case of higher-order unification) constraints are indispensable [Huet 1972, Nieuwenhuis and Rubio 1994, Vigneron 1994].

10. Effective Saturation of First-Order Theories

Saturation up to redundancy terminates for many consistent theories, provided powerful enough simplification and redundancy elimination techniques are employed. In this section we briefly describe some of the applications in which finitely saturated theories play a central role. We intend to demonstrate that saturation can be understood as a (partial) compilation process through which, when it terminates, certain objectives with regard to efficiency can be achieved in an automated way.

10.1. Decision Procedures Based on Resolution

The abstract notion of redundancy is general enough to accommodate virtually all the simplification techniques common in theorem proving. With the right setting of the resolution parameters, most of the known decidable fragments of first-order logic can be decided by saturation up to redundancy. The theory of resolution is therefore a powerful tool for decidability proofs for first-order theories and logics that can be semantically embedded in first-order logic. An early example was given by Joyner Jr. [1976] , who showed that the monadic class can be decided by ordered resolution with subsumption and condensement as simplification techniques.

Since then, many other decidable classes have been shown decidable by suitably refined calculi of ordered resolution (and paramodulation): the monadic class with equality [Bachmair, Ganzinger and Waldmann 1993], the Ackermann class with equality [Fermüller and Salzer 1993], a subclass of Maslov's class K [Fermüller, Leitsch, Tammet and Zamov 1993] and various logics for knowledge representation [Fermüller et al. 1993, Hustadt 1999]. Hustadt [1999] appears to have been the first to describe a resolution-based decision procedure for conjunctions of formulas in Maslov's class K, the completeness proof of which makes essential use of the methods presented here, in particular of renaming techniques. Schmidt [1997] proposes a general method for obtaining resolution-based decision procedures for many modal logics, and in particular provides a decision procedure for an interesting fragment of first-order logic, called *path logic*. Fermüller et al. [1993] give a comprehensive overview of earlier work on resolution-based decision methods. A recent survey of methods in this area is presented in [Fermüller, Leitsch, Hustadt and Tammet 2001] (Chapter 25 of this Handbook).

10.2. Automated Complexity Analysis

Motivated by work on "local theories" by McAllester [1993], the relation between saturation and decidability issues has been extended to complexity analysis by Basin and Ganzinger [1996], who showed that the complexity of the ordering used for saturation is directly related to the complexity of the entailment problem for a theory.

Let N be a set of standard clauses (with variables). The *(ground) entailment problem* for the *theory* N consists in checking whether or not a *query* C, where C is a variable-free standard clause, is logically implied by N. If N is saturated (up to redundancy) under standard ordered resolution without selection (that is, no atom is selected in any clause and, hence, the resolved atom is maximal in both premises), one can derive upper bounds for the complexity of the entailment problem for N. This requires an ordering on atoms in which for each ground atom A there are only finitely many smaller ground atoms. More precisely, suppose that the *complexity* of an ordering can be bounded by functions f, g in that there are at most $O(f(n))$ ground atoms smaller or equal to some ground atom in a given finite set of atoms of size n (where size refers to the number of symbols in an expression), and such that these smaller atoms can be enumerated in time $O(g(n))$.

10.1. THEOREM ([Basin and Ganzinger 1996]). *Suppose $\succ$ is a partial well-founded ordering on ground atoms of complexity f, g and N is a finite set of* Horn *clauses that is saturated under ordered resolution up to redundancy with respect to each total, well-founded extension of $\succ$. Then the entailment problem for N is decidable in time $O(f^k + g)$ where k is a constant that depends only on the theory N.*

In particular, the entailment problem is polynomial, if the ordering is of polynomial complexity.

The key technical result is expressed in the following lemma.

10.2. Lemma. *Let N be saturated up to redundancy with respect to ordered resolution (without selection) based on a total and well-founded ordering $\succ$. If C is a standard ground clause then C is a logical consequence of N if and only if C is a logical consequence of those ground instances D of N in which for each atom A in D there exists an atom B in C such that $B \succeq A$.*

Let us sketch the basic ideas underlying the lemma. If N is saturated, then inferences in which both premises are clauses in N are redundant. To decide whether some ground query C is entailed, one negates C, adds the resulting unit clauses to N, and saturates the resulting set by ordered resolution. Clauses generated by ordered inferences with one premise in N and one premise from $\neg C$ cannot generate (ground instances of) clauses in which some atom is bigger than each atom in C. Thus, if C is a logical consequence of N, it already follows from a set of ground instances of N in which all atoms are smaller than, or equal to, an atom in C. By applying dynamic programming techniques of bottom-up computation (N was assumed to be a set of Horn clauses), the result follows.

Sometimes a natural presentation of a theory is not saturated with respect to a desired ordering, but can be finitely saturated, see [Basin and Ganzinger 1996] for examples. In such cases saturation can be viewed as an optimizing compiler that adds sufficiently many "useful" consequences to a theory presentation so as to achieve a certain complexity bound for its entailment problem.

11. Concluding Remarks

The presentation of resolution theorem proving in this chapter is based on a general calculus of ordered resolution with selection for general clauses. We have described the more specialized calculi by viewing them as special cases of general resolution. Four concepts—orderings, selection functions, renaming, and redundancy—are essential in this regard. Orderings of clauses are based on well-founded, partial orderings on atoms. Slagle [1967] attributes the original idea of ordering atoms to Reynolds [1965]. Selection functions select atoms that must be true in interpretations in which the clause is false. The earliest resolution strategy that exploits selection appears to be hyper-resolution [Robinson 1965*a*]. The don't-care non-deterministic aspects of selection in resolution and the resulting pruning of resolution search spaces were first recognized by Kowalski and Kuehner [1971]. A fundamental theoretical result is the refutational completeness of this family of calculi in the presence of a certain redundancy criterion based on a well-founded ordering on formulas. The proof applies a variant of the model construction technique that was originally introduced in [Bachmair and Ganzinger 1990]. Ideas related to the notion of candidate model can be found in Brand's proof of completeness of his equality elimination method [1975] and in the work of Zhang [1988]. The definitions presented here are closely related to [Bachmair and Ganzinger 1990], though there the proofs are technically more difficult in that they deal with the equational case and with counterexample reduction and redundancy simultaneously. The paper by

Pais and Peterson [1991] contains a proof of the refutational completeness of superposition based on similar constructions, but without any mention of redundancy. Standard redundancy exploits the fact that only minimal counterexamples to the candidate model have to be reduced. Semantic properties of these partial interpretations can be exploited to further prune the search space [Ganzinger, Meyer and Weidenbach 1997]. Related ideas also play a central role in the global theorem proving method of ordered semantic hyper-linking [Plaisted and Zhu 2000]. Saturation up to redundancy is not only useful as an effective procedure to proving theorems. Refinements of resolution for specific theories can be justified on the (hypothetical) assumption that the theory be presented in a suitable saturated form.

It has often been pointed out that a weakness of resolution is its lack of goal orientation. Simplification and clause elimination based on redundancy helps ameliorate the problem, but one might also consider possible combinations of resolution with such goal-oriented methods as the sequent calculus or semantic tableaux. Avron [1993] provides some discussion of this problem. Semantic tableaux and variants thereof, including the Davis-Putnam method, model elimination and SL-resolution, can be viewed as tree-like theorem proving processes in which the limits of the individual branches are saturated under ordered resolution with selection. This view may serve as a basis for further investigations of the combination problem.

Acknowledgments

We would like to thank the reviewers of this chapter and numerous other colleagues for their helpful comments and suggestions. This research was supported in part by the National Science Foundation under grants CCR-9510072 and CCR-9902031, the German Science Foundation (Deutsche Forschungsgemeinschaft) under grants Ga 261/4-1 and Ga 261/7-1, and the ESPRIT Basic Research Working Group 22457 (Construction of Computational Logics II).

Bibliography

AVRON A. [1993], 'Gentzen-type systems, resolution and tableaux', *J. Automated Reasoning* **10**(2), 265–281.

BAAZ M., EGLY U. AND LEITSCH A. [2001], Normal form transformations, *in* A. Robinson and A. Voronkov, eds, 'Handbook of Automated Reasoning', Vol. I, Elsevier Science, chapter 5.

BACHMAIR L. AND DERSHOWITZ N. [1987], Critical pair criteria for rewriting modulo a congruence, *in* J. Davenport, ed., 'Proc. EUROCAL 87', Vol. 378 of *Lecture Notes in Computer Science*, Springer-Verlag, Berlin, pp. 452–453.

BACHMAIR L. AND GANZINGER H. [1990], On restrictions of ordered paramodulation with simplification, *in* M. Stickel, ed., 'Proc. 10th Int. Conf. on Automated Deduction, Kaiserslautern', Vol. 449 of *Lecture Notes in Computer Science*, Springer-Verlag, Berlin, pp. 427–441.

BACHMAIR L. AND GANZINGER H. [1994], Rewrite techniques for transitive relations, *in* 'Proc. 9th IEEE Symposium on Logic in Computer Science', IEEE Computer Society Press, pp. 384–393.

BACHMAIR L., GANZINGER H. AND WALDMANN U. [1993], Superposition with simplification as a decision procedure for the monadic class with equality, *in* G. Gottlob, A. Leitsch and

D. Mundici, eds, 'Proc. of Third Kurt Gödel Colloquium, KGC'93', Vol. 713 of *Lecture Notes in Computer Science*, Springer-Verlag, Berlin, pp. 83–96. Revised version of Technical Report MPI-I-93-204.

Basin D. and Ganzinger H. [1996], Complexity analysis based on ordered resolution, *in* 'Proc. 11th IEEE Symposium on Logic in Computer Science', IEEE Computer Society Press, pp. 456–465.

Baumgartner P. [1992], An ordered theory resolution calculus, *in* A. Voronkov, ed., 'Proceedings of the International Conference on Logic Programming and Automated Reasoning (LPAR'92)', Vol. 624 of *LNAI*, Springer Verlag, St. Petersburg, Russia, pp. 119–130.

Boyer R. S. [1971], Locking: A restriction of resolution, PhD thesis, University of Texas at Austin, Austin, TX.

Brand D. [1975], 'Proving theorems with the modification method', *SIAM J. Comput.* **4**, 412–430.

Bronsard F. and Reddy U. S. [1992], Reduction techniques for first-order reasoning, *in* M. Rusinowitch and J.-L. Rémy, eds, 'Conditional Term Rewriting Systems, Third International Workshop', LNCS 656, Springer-Verlag, Pont-à-Mousson, France, pp. 242–256.

Bryant R. E. [1992], 'Symbolic boolean manipulation with ordered binary-decision diagrams', *ACM Computing Surveys* **24**(3), 293–318.

Bürckert H.-J. [1990], A resolution principle for clauses with constraints, *in* M. Stickel, ed., 'Proc. 10th CADE', Vol. 449 of *Lecture Notes in Computer Science*, pp. 178–192.

Davis M., Logemann G. and Loveland D. [1962], 'A machine program for theorem proving', *Communications ACM* **5**(7), 394–397.

Davis M. and Putnam H. [1960], 'A computing procedure for quantification theory', *J. Association for Computing Machinery* **7**, 201–215.

de Nivelle H. [1996], Ordering refinements of resolution, PhD thesis, Technische Universiteit Delft.

Dershowitz N. [1987], 'Termination of rewriting', *J. Symbolic Computation* **3**, 69–116.

Fermüller C. G. and Salzer G. [1993], Ordered paramodulation and resolution as decision procedure, *in* A. Voronkov, ed., 'Proceedings of the 4th International Conference on Logic Programming and Automated Reasoning (LPAR'93)', Vol. 698 of *LNAI*, Springer Verlag, St. Petersburg, Russia, pp. 122–133.

Fermüller C., Leitsch A., Hustadt U. and Tammet T. [2001], Resolution decision procedures, *in* A. Robinson and A. Voronkov, eds, 'Handbook of Automated Reasoning', Vol. II, Elsevier Science, chapter 25.

Fermüller C., Leitsch A., Tammet T. and Zamov N. [1993], *Resolution Methods for the Decision Problem*, Vol. 679 of *Lecture Notes in Computer Science*, Springer Verlag.

Gallier J. H. [1986], *Logic for Computer Science: Foundations of Automatic Theorem Proving*, Harper and Row.

Ganzinger H., Meyer C. and Weidenbach C. [1997], Soft typing for ordered resolution, *in* 'Automated Deduction — CADE'14', Vol. 1249 of *Lecture Notes in Artificial Intelligence*, Springer-Verlag, Berlin, pp. 321–335.

Gilmore P. C. [1960], 'A proof method for quantification theory', *IBM J. Res. Develop.* **4**, 28–35.

Hsiang J. [1985], 'Refutational theorem proving using term-rewriting systems', *Artificial Intelligence* **25**, 255–300.

Huet G. [1972], Constrained Resolution: A Complete Method for Higher Order Logic, PhD thesis, Case Western Reserve University.

Hustadt U. [1999], Resolution-Based Decisison Procedures for Subclasses of First-Order Logic, PhD thesis, Fachbereich Informatik, Universität des Saarlandes, Germany.

Joyner Jr. W. [1976], 'Resolution strategies as decision procedures', *J. Association for Computing Machinery* **23**(3), 398–417.

KAPUR D. AND NARENDRAN P. [1985], An equational approach to theorem proving in first-order predicate calculus, *in* 'Proc. Ninth International Joint Conference on Artificial Intelligence', Los Angeles, CA, pp. 1146–1153.

KIRCHNER C., KIRCHNER H. AND RUSINOWITCH M. [1990], 'Deduction with symbolic constraints', *Revue Francaise d'Intelligence Artificielle* **4**(3), 9–52. Special issue on automatic deduction.

KLINGENBECK S. AND HÄHNLE R. [1994], Semantic tableaux with ordering restrictions, *in* A. Bundy, ed., 'Twelfth International Conference on Automated Deduction', LNAI 814, Springer-Verlag, Nancy, France, pp. 708–722.

KOWALSKI R. A. [1974], Predicate logic as programming language, *in* 'Proceedings of the IFIP Congress', Amsterdam, Netherlands, pp. 569–4574.

KOWALSKI R. A. AND KUEHNER D. [1971], 'Linear resolution with selection function', *Artificial Intelligence* **2**, 227–260.

LIFSCHITZ V. [1989], 'What is the inverse method?', *J. Automated Reasoning* **5**(1), 1–24.

LOVELAND D. W. [1969], 'A simplified format for the model-elimination theorem-proving procedure', *J. Association for Computing Machinery* **16**, 349–363.

LYNCH C. [1997], 'Oriented equational logic is complete', *J. Symbolic Computation* **23**(1), 23–45.

MANNA Z. AND WALDINGER R. [1980], 'A deductive approach to program synthesis', *ACM Transactions on Programming Languages and Systems* **2**(1).

MASLOV S. J. [1964], 'An inverse method for establishing deducibility in the classical predicate calculus', *Dokl. Akad. Nauk SSSR* **159**, 1420–1424.

MCALLESTER D. A. [1993], 'Automated recognition of tractability in inference relations', *J. Association for Computing Machinery* **40**(2), 284–303.

MÜLLER J. AND SOCHER-AMBROSIUS R. [1988], Topics in completion theorem proving, Research Report Memo SEKI-SR-88-13, Univ. Kaiserslautern.

MURRAY N. V. [1982], 'Completely non-clausal theorem proving', *Artificial Intelligence* **18**(1), 67–85.

NIEUWENHUIS R. AND RUBIO A. [1992], Theorem proving with ordering constrained clauses, *in* 'Automated Deduction — CADE'11', Vol. 607 of *Lecture Notes in Computer Science*, Springer-Verlag, Berlin, pp. 477–491.

NIEUWENHUIS R. AND RUBIO A. [1994], AC-superposition with constraints: No AC-unifiers needed, *in* 'Proc. 12*th* International Conference on Automated Deduction', Vol. 814 of *Lecture Notes in Computer Science*, Springer-Verlag, Berlin, pp. 545–559.

NIEUWENHUIS R. AND RUBIO A. [2001], Paramodulation-based theorem proving, *in* A. Robinson and A. Voronkov, eds, 'Handbook of Automated Reasoning', Vol. I, Elsevier Science, chapter 7.

NONNENGART A. AND WEIDENBACH C. [2001], Computing small clause normal forms, *in* A. Robinson and A. Voronkov, eds, 'Handbook of Automated Reasoning', Vol. I, Elsevier Science, chapter 6.

PAIS J. AND PETERSON G. [1991], 'Using forcing to prove completeness of resolution and paramodulation', *J. Symbolic Computation* **11**, 3–19.

PLAISTED D. AND GREENBAUM S. [1986], 'A structure-preserving clause form translation', *JSC* **2**, 293–304.

PLAISTED D. AND ZHU Y. [2000], 'Ordered semantic hyper-linking', *J. Automated Reasoning* **25**(3), 167–217.

PLOTKIN G. [1972], 'Building-in equational theories', *Machine Intelligence* **7**.

REYNOLDS J. C. [1965], Unpublished seminar notes. Stanford University.

ROBINSON J. A. [1965*a*], 'Automatic deduction with hyper-resolution', *International J. Computer Mathematics* **1**, 227–234.

ROBINSON J. A. [1965*b*], 'A machine-oriented logic based on the resolution principle', *J. Association for Computing Machinery* **12**, 23–41.

SCHMIDT R. A. [1997], Resolution is a decision procedure for many propositional modal logics, Research Report MPI-I-97-2-002, Max-Planck-Institut für Informatik, Saarbrücken, Germany.

The extended abstract version is to appear in M. Kracht, M. de Rijke, H. Wansing and M. Zakharyaschev, editors, *Advances in Modal Logic '96*, CSLI Publications 1997.

Slagle J. R. [1967], 'Automatic theorem proving with renamable and semantic resolution', *J. the ACM* **14**(4), 687–697.

Stickel M. E. [1985], Automated deduction by theory resolution, *in* A. Joshi, ed., 'Proceedings of the 9th International Joint Conference on Artificial Intelligence', Morgan Kaufmann, Los Angeles, CA, pp. 1181–1186.

Tseitin G. [1970], On the complexity of derivation in propositional calculus, *in* 'Seminars in Mathematics V.A. Steklov Math. Institute, Leningrad', Vol. 8, Consultants Bureau, New York-London, pp. 115–125. Reprinted in J. Siekmann and G. Wrightson (eds): Automation of Reasoning, Vol. 2, 1983, Springer-Verlag, pp. 466–486.

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

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

Wos L., Robinson G. and Carson D. [1965], 'Efficiency and completeness of the set of support strategy in theorem proving', *J. Association for Computing Machinery* **12**, 536–541.

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

Zhang H. [1994], 'A new method for the boolean ring based theorem proving', *J. Symbolic Computation* **17**(2), 189–212.

Index