Stratified Resolution

Anatoli Degtyarev* Robert Nieuwenhuis† Andrei Voronkov‡

October 3, 2001

Abstract

We introduce a calculus of *stratified resolution*, in which special attention is paid to clauses that "define" relations. If such clauses are discovered in the initial set of clauses, they are treated using the rule of *definition unfolding*, i.e. the rule that replaces defined relations by their definitions. Stratified resolution comes with a powerful notion of redundancy: a clause to which definition unfolding has been applied can be removed from the search space. To prove the completeness of stratified resolution with redundancies, we use a novel combination of Bachmair and Ganzinger's model construction technique and a hierarchical construction of orderings and least fixpoints.

Contents

---

*Manchester Metropolitan University
†Technical University of Catalonia
‡University of Manchester

1 Introduction

Most state-of-the-art theorem provers in first-order logic apply some variant of ordered resolution with *selection*: in each clause a subset of its literals is selected, and these selected literals are the only active ones in resolution inferences with this clause. The selection strategy (deciding which literals are selected) is usually parametrized by a given ordering on ground atoms.

For several of these strategies refutational completeness is known. For example, in the ground case, it suffices to select in each clause either a negative literal or else some positive atom that is maximal with respect to the given ordering among all atoms of the clause. In the nonground case, one can select either a negative literal or else a subset of the positive literals such that for all ground instances the maximal literal is among the selected ones [Bachmair and Ganzinger 2000].

In the Horn case, completeness is preserved if a single arbitrary literal in each clause is selected. For example, Prolog's SLD-resolution amounts to selecting the positive literal whenever there is any, which, very conveniently, converts the search process into a form of goal-directed definition unfolding.

In this article we show that, under some precisely defined circumstances, it is also complete to do this in the nonHorn case. We introduce *stratified resolution*, a resolution calculus with special rules for handling hierarchical definitions of relations. Stratified resolution generalizes SLD-resolution to a more general case, where clauses may be nonHorn but "Horn with respect to a set of *defined* relations". In this calculus, one can select a single positive literal $p(\ldots)$ of a (possibly nonHorn) clause C, i.e., only use C for unfolding the definition of p, even if this atom $p(\ldots)$ is not strictly larger than the atoms occurring negatively in C. Consider the following (for explanation purposes, naively simple) propositional example:

Example 1.1 Assume we have, among others, the two clauses $p \vee \neg q \vee r$ and $q \vee \neg p$. Then in our calculus it is complete to consider both these clauses as definitions of their positive literals p and q, i.e., selecting these positive literals, which does not follow from the standard results for ordered resolution with selection. $\square$

Making the right choices in selection is crucial for the performance of a deduction process. Consider the following example.

Example 1.2 Suppose we try to establish the inconsistency of a set of clauses S containing a recursive definition of a relation *split* that splits a list of conferences into two sublists: of deduction-related conferences, and of all other conferences.

$$\begin{array}{l} split(\texttt{[}x\,|\,y\texttt{]},\texttt{[}x\,|\,z\texttt{]},u) \texttt{ :- } deduction(x), split(y,z,u).\\ split(\texttt{[}x\,|\,y\texttt{]},z,\texttt{[}x\,|\,u\texttt{]}) \texttt{ :- } \neg deduction(x), split(y,z,u).\\ split(\texttt{[]},\texttt{[]},\texttt{[]}). \end{array}$$

Suppose that S also contains other clauses, for example

$$\neg split(x,y,z) \vee conference\_list(x).$$

and assume our ordering on atoms is primarily based on an ordering on their predicate symbols. Now we face several choices in selecting the order and negative literals in clauses. For example, if we make *deduction* greater than *split*, then we must select either $\neg deduction(x)$ or $\neg split(y,z,u)$ in the first clause. It seems much more natural to select $split(\texttt{[}x\,|\,y\texttt{]},\texttt{[}x\,|\,z\texttt{]},u)$ instead, then we can use the first clause in the same way it would be used in logic programming. Likewise, if we always try to select a negative literal in a clause, the literal $\neg split(y,z,u)$ will be selected in the second clause, which is most likely a wrong choice, since then any resolvent with the second clause will give us a larger clause.

Let us now choose an ordering in which the literals $\mathit{split}(\texttt{[}x\,|\,y\texttt{]}, \texttt{[}x\,|\,z\texttt{]}, u)$ and $\mathit{split}(\texttt{[}x\,|\,y\texttt{]}, z, \texttt{[}x\,|\,u\texttt{]})$ are maximal in their clauses, and select these literals. Consider the fourth clause. If we select $\neg \mathit{split}(x, y, z)$ in it, we can resolve this literal with all three clauses defining *split*. It would be desirable to select $\mathit{conference\_list}(x)$ in it, since a resolvent upon $\neg \mathit{conference\_list}(x)$ is likely to instantiate x to a nonvariable term t, and then the literal $\neg \mathit{split}(t, y, z)$ can be resolved with only two, one or no clauses at all, depending on the form of t.

In all cases, it seems reasonable to choose an ordering and selection function in such a way that the first three clauses will be used as a *definition* of *split* so that we *unfold* this definition, i.e., replace the heads of these clauses with their bodies. Such an ordering would give us the best results if we have an adequate strategy of negative selection which says: select $\neg \mathit{split}(r, s, t)$ only if r is instantiated enough, or if we have no other choice. □

In order to implement this idea we have to be able to formalize the right notion of "definition" in a set of clauses. Such a formalization is undertaken in this article, in the form of a calculus of *stratified resolution*. Stratified resolution is based on the following ideas that can be tracked down to earlier ideas developed in logic programming.

1. Logic programming is based on the idea of using definite clauses as definitions of relations. Similar to the notion of definite clause, we introduce a more general notion of a set of clauses *definite w.r.t. a set of relations*. These relations are regarded as defined by this set of clauses.
2. In logic programming, relations are often defined in terms of other relations. The notion of *stratification* [Gelder 1988, Apt and Blair 1988, Przymusinski 1988] allows one to formalize the notion "P is defined in terms of Q". We use a similar idea of stratification, but in our case stratification must be related to a reduction ordering $\succ$ on literals.

The difficult problem is to find automatically the right ordering that makes the atom in the head of a "definition" greater than atoms in the body of this definition. Consider, for example, clauses defining reachability in a directed graph.

EXAMPLE 1.3 Assume a graph is formalized by the binary relation *edge*. The reachability can be defined by

$$\begin{array}{l} \mathit{reachable}(x,y) \texttt{ :- } \mathit{edge}(x,y). \\ \mathit{reachable}(x,z) \texttt{ :- } \mathit{edge}(x,y), \mathit{reachable}(y,z). \end{array}$$

There is no well-founded ordering stable under substitutions that makes the atom $\mathit{reachable}(x, z)$ greater than $\mathit{reachable}(y, z)$. So the standard ordered resolution with negative selection cannot help us in selecting the "right" ordering. The theory developed in this paper allows one to select only the literal $\mathit{reachable}(x, z)$ in this clause despite that this literal is not the greatest. □

In addition to intelligent selection of literals in clauses, certain notions of redundancy can be exploited in many more cases in stratified resolution than in standard ordered resolution calculi. To explain this kind of redundancy, let us come back to Example 1.2.

EXAMPLE 1.4 (Example 1.2 continued) Suppose we have a clause

$$\neg \mathit{split}(\texttt{[}\mathit{cade}, \mathit{www}, \mathit{lpar}\texttt{]}, y, z). \tag{1}$$

Stratified resolution can resolve this clause with the first two clauses in the definition of *split*, obtaining two new clauses

$$\neg deduction(cade) \vee \neg split([www, lpar], y, z)$$
$$deduction(cade) \vee \neg split([www, lpar], y, z)$$

In general, these two clauses would be *added* to the search space. However, if one can ensure that no more inferences will be needed again on (1), then they can as well *replace* clause (1) thus making the search space smaller. Indeed, in stratified resolution this is the case for all "defined" predicates like *split*, and hence this situation is far more frequent, and more easily detectable, than in standard ordered resolution, where this situation applies only if in all clauses where *split* occurs in a positive literal, it occurs only once positively, and it is the only selected literal in its clause. □

When the initial set of clauses contains no definitions or cannot be stratified, stratified resolution becomes ordinary ordered resolution with negative selection. However, sets of clauses which contain definitions and can be stratified in our sense are often met in practice, since they correspond to a frequently used form of (possibly recursive) definitions of relations. For example, a majority of TPTP problems can be stratified.

This article is organized as follows. After the basic notions and notations of Sections 2 and 3, in Section 4 we first define the ground version of stratified resolution and prove its completeness. Then, in its Subsections 4.1 and 4.2 we define stratified resolution derivations with redundancies, and we show that a clause can be removed from the search space after definition unfolding has been applied to it. Then in Section 5 we explain the full nonground version of stratified resolution with redundancy.

In this article we deal with first-order logic without equality. We will only briefly discuss equality in Section 8.

Related work

There are not many papers in the automated deduction literature relevant to this work. Our formal system resembles SLD-resolution [Kowalski and Kuehner 1971]. When the initial set of clauses is Horn, our stratified resolution with redundancies becomes SLD-resolution.

The possibility of arbitrary selection for Horn clauses follows from a result of [de Nivelle 1996]: if a set of clauses has a resolution refutation without factoring, then it has a refutation with an arbitrary selection of literals in clauses.

A first version [Degtyarev and Voronkov 2000] of this work appeared in CADE-17, using a proof technique based on *traces*, which may be interesting by itself. But the definitions of stratification as well as the proofs given here are significantly simpler, and allow us to answer here some of the problems left open in [Degtyarev and Voronkov 2000].

2 Basic notions

Let $\succ$ be a (strict partial) ordering (i.e., a transitive and irreflexive binary relation). The *multiset extension* of $\succ$, denoted by $\succ^{mul}$, is defined as the smallest transitive relation on finite multisets such that

$$X \cup \{A\} \succ^{mul} X \cup \{B_1, \ldots, B_n\} \qquad \text{if } A \succ B_i \text{ for all } i \in \{1 \ldots n\}$$

If $\succ$ is a well-founded ordering on S then $\succ^{mul}$ is a well-founded ordering on finite multisets of elements of S.

A *quasi-ordering* is a transitive and reflexive binary relation, denoted in this article by $\succeq$ (possibly with subscripts or superscripts), and its inverse is denoted by $\preceq$. Its *strict part*, denoted by $\succ$, is the strict ordering $\succeq \setminus \preceq$ (i.e, $s \succ t$ iff $s \succeq t$ and $s \not\preceq t$). Its *equivalence*, denoted by $\sim$, is defined as $\succeq \cap \preceq$. Note that $\succeq$ is the disjoint union of $\succ$ and $\sim$.

In this article we will deal with several orderings on the set of ground atoms of a signature Σ. For *every* such ordering $\succ$, let $\succ_{lit}$ be the smallest extension of $\succ$ to the set of ground literals of Σ such that for all ground atoms R and R' we have (i) $\neg R' \succ_{lit} \neg R$ whenever $R' \succ R$, (ii) $\neg R \succ_{lit} R$ and (iii) $\neg R \succ_{lit} L \succ_{lit} R$ for no L. To keep notation simple, we will omit the subscript *lit* and use the same notation for $\succ$ and $\succ_{lit}$. This causes no ambiguity, since $\succ$ and $\succ_{lit}$ agree on ground atoms.

A *clause* is a multiset of literals. The *empty clause* is denoted by $\square$. We assume knowledge of substitutions, unifiers, and most general unifiers. A most general unifier of two atoms (or literals) A and B will be denoted by $mgu(A, B)$. The application of a substitution σ to an expression (eg., an atom, literal, or clause) E is denoted by $E\sigma$. An expression is *ground* if no variable occurs in it. A *ground instance* of any expression E is an expression $E\sigma$ which is ground. The set of all ground instances of any expression E will be denoted by $gnd(E)$, and if S is a set of clauses, by $gnd(S)$ we denote the set of ground instances of clauses in S, that is, $gnd(S) = \{C \mid C \in gnd(D) \text{ and } D \in S\}$.

We consider (partial) Herbrand interpretations as boolean functions over a set of ground atoms $\mathcal{A}$, that is, functions $I : \mathcal{A} \to \{0, 1\}$. I is called *total* if $\mathcal{A}$ is the set of all ground atoms over the given signature, and *partial* otherwise. I is said to be *defined* for the ground atoms, ground literals and ground clauses built over $\mathcal{A}$. A ground atom $R \in \mathcal{A}$ is called *true* in I if and $I(R) = 1$ and *false* otherwise. This notion of truth and falsehood is extended in the usual way to ground literals and ground clauses built over $\mathcal{A}$. A clause C is true in I if I is defined for all ground instances of C, and all these instances are true in I. The interpretation I is a *model* of a set a clauses S if all clauses of S are true in I. Finally, if E is an atom, literal, or clause such that E is true in I, then we write $I \models E$, and we write $I \not\models E$ if E is false in I.

3 Stratifications

In this article, we assume a finite signature Σ whose set of predicate symbols is the disjoint union of two sets: the so-called *defined* symbols $\mathcal{P}$ and the remaining *nondefined* ones $\mathcal{Q}$. In what follows, (possibly indexed) p and q always denote elements of $\mathcal{P}$ and $\mathcal{Q}$ respectively, and r denotes an arbitrary symbol of $\mathcal{P} \cup \mathcal{Q}$.

DEFINITION 3.1 A $\mathcal{P}$-atom is an atom of the form $p(t_1, \ldots, t_n)$ with $p \in \mathcal{P}$. A $\mathcal{Q}$-atom is an atom of the form $q(t_1, \ldots, t_n)$ with $q \in \mathcal{Q}$. A $\mathcal{P}$-literal (respectively $\mathcal{Q}$-literal) is a possibly negated $\mathcal{P}$-atom (respectively $\mathcal{Q}$-atom). We denote $\mathcal{P}$-atoms by P, $\mathcal{Q}$-atoms by Q, and arbitrary atoms by R, possibly with indices.

We assume a fixed total quasi-ordering $\succeq_{pred}$ on $\mathcal{P} \cup \mathcal{Q}$ such that $r \sim_{pred} r'$ only if both r and r' are in $\mathcal{P}$. We also assume a total fixed quasi-ordering $\succeq$ on the set of ground atoms of Σ which *respects* $\succeq_{pred}$, i.e., $r(s_1, \ldots, s_m) \succeq r'(t_1, \ldots, t_n)$ whenever $r \succeq_{pred} r'$ and r and r' are different or defined.

Note that the ordering $\succeq$ on ground atoms is determined by the ordering $\succeq_{pred}$ on their head symbols, except for $\mathcal{Q}$-atoms headed with the same symbol, which can be ordered in any way such that $\succ$ is well-founded.

Definition 3.2 (definition clauses) A set S of clauses over $\mathcal{P} \cup \mathcal{Q}$ is *Horn with respect to* $\mathcal{P}$ if no clause in S contains more than one positive $\mathcal{P}$-literal. Clauses containing exactly one positive $\mathcal{P}$-literal are called *definition clauses* or simply *definitions*.

The following definition is central:

Definition 3.3 (stratification) A set S of clauses is *stratified* w.r.t. $\succeq$ if it is Horn with respect to $\mathcal{P}$ and in every definition clause $P \vee C$ in S, it holds that $P \succeq R$ for all atoms R occurring (positively or negatively) in C. Then $\succeq$ together with the splitting of the set of predicate symbols into $\mathcal{P}$ and $\mathcal{Q}$ is called a *stratification* for S. Note that the relations $P \succeq R$ are fully determined by the ordering $\succeq_{pred}$ on the head symbols of P and R.

Example 3.4 Consider the set consisting of four clauses: $r_1 \vee r_2$, $r_1 \vee \neg r_2$, $\neg r_1 \vee r_2$, and $\neg r_1 \vee \neg r_2$. This set is Horn with respect to $\{r_1\}$ and also with respect to $\{r_2\}$, but not with respect to $\{r_1, r_2\}$. To stratify this set of clauses, we can e.g. use the ordering $r_1 \succ r_2$ (i.e., r_1 is considered as a relation defined in terms of r_2), This example shows that in general there is no greatest set of defined clauses.

Definition 3.5 (stratified selection function) A *selection function* is any function *sel* on the set of clauses such that (i) $sel(C)$ is a submultiset of C, (ii) $sel(C)$ is nonempty whenever C is nonempty, and (iii) either $sel(C)$ contains only positive literals, or $sel(C)$ contains exactly one negative literal. If $L \in sel(C)$, we say that L is *selected by sel* in C.

We say that a selection function is *stratified* if it also satisfies the following conditions: (iv) in any definition clause $P \vee C$, only P is selected; (v) in any nondefinition clause C with a positive selected literal a subset $\{R_1, \ldots, R_n\}$ of its positive literals is selected such that C is of the form $R_1 \vee \ldots \vee R_n \vee D$, and for all ground instances $C\sigma$ and every atom B in D, it holds that $R_i\sigma \succ B\sigma$ for some i.

In the following, we will assume a fixed stratified selection function. We underline selected literals, so when we write $\underline{A} \vee C$, this means that A (and maybe some other literals) are selected in $A \vee C$.

4 Stratified resolution: the ground case

Here we introduce ground stratified resolution and illustrate our proof techniques in the ground case. In this section we only work with ground clauses.

Definition 4.1 The inference system of ground *stratified resolution* consists of two inference rules:

1. *ground stratified resolution rule:*

$$\frac{\underline{R} \vee C \quad \underline{\neg R} \vee D}{C \vee D}, \quad \text{where if } R \text{ is a } \mathcal{Q}\text{-atom, then for every literal } L \text{ in } C \text{ we have } R \succ L.$$

2. *ground positive factoring rule:*

$$\frac{\underline{Q} \vee \underline{Q} \vee C}{Q \vee C}, \quad \text{where for every literal } L \text{ in } C \text{ we have } Q \succeq L.$$

Note that the stratified resolution rule is defined as the standard rule of binary resolution with selection, but we use a stratified selection function, and that in the positive factoring rule Q is a $\mathcal{Q}$-atom.

In the previous stratified resolution rule, the clause $\underline{\neg R} \vee D$ is sometimes called the *rightmost premise*, and in the positive factoring rule $\underline{Q} \vee \underline{Q} \vee C$ is the (only, and hence) rightmost premise.

The choice of which symbols are in $\mathcal{P}$ and which ones are in $\mathcal{Q}$, as well as the choice of the ordering $\succeq$ will be crucial for finding stratifications that reduce the search space in practical theorem provers. In general, good choices lead to stratifications with larger sets $\mathcal{P}$, in order to minimize the amount of inferences and maximize the amount of redundancy (see Examples 1.1, 1.2, 1.3, 3.4, and 4.5).

The following proposition states that stratification is preserved under our inference rules. This holds since stratification is a property depending on the subset of definition clauses, and stratified resolution does not introduce any new definition clauses.

PROPOSITION 4.2 *Let $\succeq$ be a stratification for a set of clauses S. Let C be the conclusion of an inference in the system of stratified resolution with premises in S. Then $\succeq$ is also a stratification for $S \cup \{C\}$. Moreover, C is not a definition clause.* □

THEOREM 4.3 *Ground stratified resolution is refutation complete for stratified sets of ground clauses.*

We do not prove this theorem now; it will follow from the more general Theorem 4.8 proved below. In our completeness proofs we will use a model construction formalized in the following definition of a *stratified interpretation.*

DEFINITION 4.4 (stratified interpretation) Let S be a set of ground clauses that is stratified w.r.t. $\succeq$. The *stratified interpretation* I of S w.r.t. $\succeq$ is defined by induction on $\succ$. To define the value of I on an atom R, we will use the partial interpretation defined in this construction for all ground atoms R' with $R \succ R'$, i.e. the restriction of I on $\{R' \mid R \succ R'\}$, denoted $I_{\prec R}$. The value $I(R)$ is defined in terms of $I_{\prec R}$ as follows:

1. For $\mathcal{Q}$-atoms Q, we define $I(Q) = 1$ if, and only if, there is some clause $\underline{Q} \vee C$ in S such that (i) $Q \succ R$ for all atoms R in C, and (ii) $I_{\prec Q} \not\models C$ (note that $I_{\prec Q}$ is defined for C due to condition i).

2. For $\mathcal{P}$-atoms P, let $\mathbb{P}$ denote the set $\{P' \mid P \sim P'\}$, and let S_P be the set of all definition clauses in S of the form
$$\underline{P'} \vee \neg P_1 \vee \ldots \vee \neg P_n \vee C \tag{2}$$
such that (i) $P \sim P' \sim P_i$ for all i in $\{1 \ldots n\}$, (ii) $P \succ R$ for all atoms R in C (again, note that hence $I_{\prec P}$ is defined for C), and (iii) $I_{\prec P} \not\models C$.

   Now, in a similar way as done in the iterated fixpoint construction of the perfect model for stratified logic programs (see e.g. [Apt 1990]), define a sequence $\mathbb{P}_0, \mathbb{P}_1, \ldots$ of subsets of $\mathbb{P}$ by induction as follows:

   - $\mathbb{P}_0 = \{P' \in \mathbb{P} \mid$ there exists a clause of the form (2) in S_P such that n=0$\}$
   - For $i \geq 0$ define $\mathbb{P}_{i+1} = \mathbb{P}_i \cup \{P' \in \mathbb{P} \mid$ there exists a clause of the form (2) such that $\{P_1, \ldots, P_n\} \subseteq \mathbb{P}_i\}$.

Finally, for every $P' \in \mathbb{P}$ we define $I(P') = 1$ iff P' is in $\bigcup_i \mathbb{P}_i$. In that case we say that $P' \in \mathbb{P}$ has *level* k, denoted $level(P) = k$, if k is the smallest number such that $P' \in \mathbb{P}_k$. We denote by $\succ_S$ the smallest ordering on ground atoms extending $\succ$ such that $P \succ_S P'$ if $P \sim P'$, both P and P' have levels, and $level(P) > level(P')$. It is not hard to argue that $\succ_S$ is well-founded. $\square$

Note that $\succ_S$ in general depends on S, but for all S we have that $\succ_S$ is an extension of $\succ$. This implies that $\succ_S^{mul}$ is an extension of $\succ^{mul}$.

EXAMPLE 4.5 Let us now show that the ordering condition on the definition clauses $P \vee C$, namely that $P \succeq R$ for all atoms R occurring in C, is essential for completeness. We show that violation of this conditions causes incompleteness even when R is a $\mathcal{Q}$-atom. This example is taken from [Lynch 1997]. Consider the following set of propositional clauses:

$$\begin{array}{lll} \neg q \vee \underline{r} & \neg p \vee \underline{q} & \neg r \vee \underline{\neg q} \\ \neg q \vee \underline{\neg p} & \neg p \vee \underline{\neg r} & \\ \neg r \vee \underline{p} & r \vee q \vee \underline{p} & \end{array}$$

This clause is unsatisfiable and Horn w.r.t. $\{p\}$. Consider the ordering $r \succ q \succ p$. This ordering violates the ordering condition on the definition clauses $\neg r \vee \underline{p}$ and $r \vee q \vee \underline{p}$. The empty clause cannot be derived from it, even if tautologies are allowed. Indeed, the conclusion of any inference by stratified resolution is subsumed by one of the clauses in this set. $\square$

4.1 Redundancy in the ground case

The following notions are adaptations of well-known concepts for redundancy in saturation-based first-order theorem proving (see [Bachmair and Ganzinger 2000]).

There is a significant difference, however: apart from relying on the atom ordering $\succ$, our clause ordering $\succ_S^{mul}$ is based on its extension $\succ_S$, which depends on the level information of the fixpoint construction for the given set S of clauses. For the completeness proofs to go through, we will use the ordering $\succ_S$. But as for practical use of redundancy criteria, we can only use its approximation $\succ$ since $\succ_S$ can be undecidable and even not recursively enumerable.

Definition 4.6 below, defining the redundancy of inferences w.r.t. a given fixed set S and defining *saturatedness* of S, is given in terms of the ordering $\succ_S^{mul}$. In Definition 4.10, defining redundancy of clauses in stratified resolution derivations, the approximation $\succ^{mul}$ of $\succ_S^{mul}$ is used instead.

DEFINITION 4.6 Let S be a set of ground clauses, C be a ground clause, and $>$ be any ordering on ground clauses. Denote $S^{<C} = \{C' \in S \mid C > C'\}$. We will use this definition with $>$ either equal to $\succ^{mul}$ or $\succ_S^{mul}$.

1. An inference (by stratified resolution or positive factoring) with the rightmost premise C and conclusion D is *redundant w.r.t. S and $>$* if $S^{<C} \models D$.

2. S is *saturated* if for every inference (by stratified resolution or positive factoring) with premises in S, either (i) the rightmost premise of the inference is not larger w.r.t. $\succ_S^{mul}$ than the conclusion, or (ii) the inference is redundant w.r.t. S and $\succ_S^{mul}$.

EXAMPLE 4.7 (saturated set) Consider the quasi-ordering $\succeq$ such that

$$r \succ p_1 \sim p_2 \succ q$$

and the set S of clauses

$$\begin{array}{lll} 1.\ \underline{p_1} \vee \neg p_2 & 2.\ \underline{p_1} \vee q & 3.\ \underline{p_2} \vee \neg p_1 \\ 4.\ \underline{\neg p_1} \vee r & 5.\ \underline{r} \vee q & \end{array}$$

In the stratified interpretation of S the atoms p_1, p_2, r are true and q is false. The atom p_1 becomes true in this interpretation due to clause 2, p_2 due to clause 3, and r due to clause 5. This interpretation satisfies all the clauses. The ordering $\succ_S$ extends $\succ$ by $p_2 \succ_S p_1$ because the level of p_2 is greater than the level of p_1.

The set S is saturated. Indeed, the resolution inference between clauses 1 and 4 satisfies condition (i) of the definition of a saturated set. Likewise, the resolution inference between clauses 2 and 4 is redundant because the conclusion $q \vee r$ follows from clause 5, and 5 is smaller than the rightmost premise $\underline{\neg p_1} \vee r$. $\square$

Note that in practice one usually cannot exploit case (i) of the definition of saturatedness, nor the actual ordering $\succ_S^{mul}$ for case (ii). Instead, a practical sufficient condition ensuring that a set of clauses S is saturated is that all inferences with premises in S are redundant w.r.t. S and $\succ^{mul}$.

THEOREM 4.8 *Let S be a saturated stratified set of ground clauses. Then $\square \in S$ whenever S is unsatisfiable.*

PROOF. Suppose, by contradiction, $\square \notin S$. It suffices to show that then S is satisfiable. We do this by actually exhibiting a model of S, namely I, the stratified interpretation (see Definition 4.4) of S w.r.t. $\succeq$. We will use some notation of Definition 4.4 in the proof.

We now show $I \models S$ by contradiction: assume I is not a model of S. Then there is a clause $C \in S$ which is false in I. Let C be a *minimal* w.r.t. $\succ_S^{mul}$ clause such that $I \not\models C$ (such a clause exists since $\succ_S^{mul}$ is well-founded).

Consider four cases depending on the set of selected literals of C.

1. *A positive $\mathcal{Q}$-literal is selected*, i.e., C has the form $\underline{Q} \vee D$. By our definition of stratified selection function, we can assume that for all literals $L \in D$ we have $Q \succeq L$. Now we distinguish two cases, depending on whether $Q \in D$ or not. If $Q \in D$, then (again by our notion of selection) C is of the form $\underline{Q} \vee \underline{Q} \vee D'$ for some D'. Then, consider the inference by positive factoring:

$$\frac{\underline{Q} \vee \underline{Q} \vee D'}{Q \vee D'}$$

   Since S is saturated, and the conclusion $Q \vee D'$ is smaller w.r.t. $\succ_S^{mul}$ than the premise, the inference must be redundant in S w.r.t. $\succ_S$, that is, the conclusion $Q \vee D'$ follows from clauses in S strictly smaller w.r.t. $\succ_S^{mul}$ than the premise $\underline{Q} \vee \underline{Q} \vee D'$. But then, since $Q \vee D'$ is false in I, at least one of these smaller clauses is false too in I which contradicts the minimality of $\underline{Q} \vee \underline{Q} \vee D'$.

   If $Q \notin D$, and hence for all literals $L \in D$ we have $Q \succ L$, then by our construction $I(Q) = 1$ and therefore $I \models Q$, which contradicts $I \not\models C$.

2. *A positive $\mathcal{P}$-literal is selected*, i.e., C has the form $\underline{P} \vee \neg P_1 \vee \neg P_n \vee D$, where $P \sim P_i$ for all $i \in \{1, \ldots, n\}$ and $P \succ L$ for all atoms L occurring in D. Since $I \not\models C$, then $I \not\models D$ and $I \models P_i$ for all $i \in \{1, \ldots, n\}$. By the construction of I this implies that $I \models P$, which contradicts $I \not\models C$.

3. *A negative $\mathcal{Q}$-literal is selected*, i.e., C has the form $\underline{\neg Q} \vee D$. Then $I \models Q$ and by construction of I then there is some clause $\underline{Q} \vee D'$ in S such that Q is strictly greater than all literals in D' w.r.t. $\succ$ and $I \not\models D'$. Then Q is also strictly greater than all literals in D' w.r.t. $\succ_S$. Consider the inference by stratified resolution

$$\frac{\underline{Q} \vee D' \quad \underline{\neg Q} \vee D}{D' \vee D}$$

whose conclusion $D' \vee D$ is false in I and smaller w.r.t. $\succ_S^{mul}$ than the rightmost premise $\underline{\neg Q} \vee D$. Since S is saturated, this inference is redundant w.r.t. S and $\succ_S^{mul}$, so the conclusion $D' \vee D$ of this inference must follow from clauses in S strictly less w.r.t. $\succ_S^{mul}$ than $\underline{\neg Q} \vee D$. But then, since $D' \vee D$ is false in I, at least one of these clauses is false too in I which contradicts the minimality of $\underline{\neg Q} \vee D$.

4. *A negative $\mathcal{P}$-literal is selected*, i.e., C has the form $\underline{\neg P} \vee D$. Then $I \models P$. Hence by construction of I, P has some level l. Again by the construction, there exists some clause $\underline{P} \vee \neg P_1 \vee \ldots \vee \neg P_n \vee D'$ in S where $P \sim P_i$ for all $i \in \{1, \ldots, n\}$, where $P \succ L$ for all atoms L occurring in D', and $P_1, \ldots, P_n$ have levels strictly smaller than l. This implies $P \succ_S P_i$ for all $i \in \{1, \ldots, n\}$. Then P is strictly greater w.r.t. $\succ_S$ than all literals in $\neg P_1 \vee \ldots \vee \neg P_n \vee D'$. By repeating the argument of the previous case with an inference of stratified resolution between $\underline{\neg P} \vee D$ and $\underline{P} \vee \neg P_1 \vee \ldots \vee \neg P_n \vee D'$, the false conclusion $\neg P_1 \vee \ldots \vee \neg P_n \vee D' \vee D$ follows from clauses in S strictly less w.r.t. $\succ_S^{mul}$ than $\underline{\neg P} \vee D$, and so we obtain a contradiction.

$\square$

As said, in the following definition, where one considers derivations in which the sets of clauses dynamically change, we define redundancy of clauses and fairness in terms of the approximation ordering $\succ^{mul}$.

Definition 4.9 Let C be a ground clause, let $>$ be an ordering on ground clauses, and let S be a set of clauses. Then C is called *redundant w.r.t. S and $>$* if $S^{<C} \models C$.

Definition 4.10

1. A *stratified resolution derivation* is a sequence of sets of clauses $S_0, S_1, \ldots$ such that S_0 is stratified, and each S_{i+1} is obtained from S_i either by adding to S_i a logical consequence of S_i that is not a definition or by removing from S_i some clause that is redundant w.r.t. S_i and $\succ^{mul}$.

2. A clause is *persistent* in the derivation if, for some j, it belongs to all S_k with $k \geq j$.

3. The derivation is *fair* (for the given inference system) if, for every inference with persistent premises, there exists some S_j such that either the inference is redundant w.r.t. S_j and $\succ^{mul}$, or else its conclusion belongs to S_j.

Theorem 4.11 *Let $S_0, S_1, \ldots$ be a fair ground stratified resolution derivation. Then $\square \in S_j$ for some j if, and only if, S_0 is unsatisfiable.*

Proof. It is not hard to argue that each clause occurring in the derivation is a logical consequence of S_0, hence $\square \in S_j$ implies that S_0 is unsatisfiable.

To prove the "if" direction, suppose that S_0 is unsatisfiable. Denote the set of persistent clauses by S. Using well-foundedness of $\succ^{mul}$ one can prove that every clause removed from the derivation is a logical consequence of smaller w.r.t. $\succ^{mul}$ clauses in S. This implies that S is logically equivalent to S_0, and hence S is unsatisfiable. Since S is stratified (no new definition clauses are generated in the derivation), if we now prove that S is saturated, then by Theorem 4.8 we obtain $\square \in S$ and hence $\square \in S_j$ for some j. To this end consider any inference by stratified resolution with premises in S (the case of factoring is similar):

$$\frac{\underline{R} \lor C \quad \underline{\neg R} \lor D}{C \lor D}$$

For proving that S is saturated, we have to show that either (i) the rightmost premise $\underline{\neg R} \lor D$ of the inference is not larger w.r.t. $\succ_S^{mul}$ than the conclusion $C \lor D$, or (ii) the inference is redundant w.r.t. S and $\succ_S^{mul}$.

To this end we assume that (i) does not hold, i.e., we assume $\underline{\neg R} \lor D \succ_S^{mul} C \lor D$, and show (ii). Since the derivation is fair, there exists some S_j such that either the inference is redundant w.r.t. S_j and $\succ^{mul}$, or else its conclusion $C \lor D$ belongs to S_j. Note that, since $\succ_S^{mul}$ extends $\succ^{mul}$, in both cases the conclusion $C \lor D$ follows from clauses in S_j smaller w.r.t. $\succ_S^{mul}$ than $\underline{\neg R} \lor D$.

But each one of these smaller clauses in S_j follows from persistent clauses that are again smaller or equal w.r.t. $\succ^{mul}$ (and hence w.r.t. $\succ_S^{mul}$). Therefore, $C \lor D$ follows from persistent clauses that are smaller w.r.t. $\succ_S^{mul}$ than $\underline{\neg R} \lor D$, i.e., the inference is redundant w.r.t. S and $\succ_S^{mul}$. $\square$

4.2 Deletion of unfolded clauses

There are standard ways for computing fair derivations in practice; for example, the clauses can be stored in some (priority) queue ensuring that every inference with such clauses is eventually either proved redundant or else its conclusion is added. A simpler possibility used in most provers is to only remove clauses redundant w.r.t. the current S_i instead of using redundancy w.r.t. all S_j.

However, with stratified resolution one can remove some clauses that do not follow from smaller ones. This is based on the observation that no new definition clauses are generated during derivations, so no clause $\underline{\neg P} \lor C$ resolved against all definition clauses of the form $P \lor D$ can participate in any new inference. Thus, we can delete $\underline{\neg P} \lor C$ at the point of the derivation when it has already been resolved with all definitions available at this point.

This can be formalized as follows. Let us change the notion of derivation by adding a new deletion rule, called *deletion of unfolded clauses.* Suppose that S_i contains a clause $\underline{\neg P} \lor C$, and for every definition clause $\underline{P} \lor D$ in S_i, either the resolvent $D \lor C$ of these two clauses belongs to some S_j for $j \leq i$, or else the corresponding inference is redundant w.r.t. $\succ^{mul}$ and some S_j for $j \leq i$. Then $\underline{\neg P} \lor C$ can be deleted from S_i.

Let us call the resulting inference system *stratified resolution with deletion of unfolded clauses.* The notion of fair derivation in this system remains as before.

Theorem 4.12 *Stratified resolution with deletion of unfolded clauses is complete, i.e., every fair derivation from an unsatisfiable set S_0 of clauses contains the empty clause.*

Proof. Let $S_0, S_1, \ldots$ be such a derivation. Consider the derivation $S_0, S_1', S_2', \ldots$ obtained from $S_0, S_1, \ldots$ by keeping all deleted unfolded clauses. It is not hard to argue that $S_0, S_1', S_2', \ldots$ is fair, and hence some S'_i contains the empty clause. Since the empty clause cannot be deleted, it is also contained in some S_j. □

In a practical theorem prover an unfolded clause $\underline{\neg p} \vee D$ can be either deleted or blocked for further inferences, although it can still be used in redundancy proofs of other clauses or inferences.

5 Nonground clauses

In this section we extend the results of the previous section to the case of nonground clauses.

No ordering on nonground items is used. We keep the ordering $\succeq_{pred}$ and, on the set of ground atoms, the total fixed quasi-ordering $\succeq$ respecting $\succeq_{pred}$ (remember: $\succeq$ is determined by $\succeq_{pred}$, except for $\mathcal{Q}$-atoms headed with the same symbol, which can be ordered in any way by $\succeq$ as long as $\succ$ is well-founded). In a practical prover usually general-purpose orderings like the Knuth-Bendix ordering (KBO) or the lexicographic or recursive path ordering (LPO or RPO) can be used for ordering such $\mathcal{Q}$-atoms (see e.g., [Niewenhuis and Rubio 2000] for precise definitions).

In practice, for selection functions on nonground clauses (and, as we will see, for restricting more the nonground inference rules), it will be useful to be able to approximate $\succeq$ on the nonground level, that is, to check, for nonground $\mathcal{Q}$-atoms Q and Q', whether there exists some ground substitution σ such that $Q\sigma \succ Q'\sigma$, or, more generally, whether for a boolean formula F over relations $Q \succ Q'$ or $Q \succeq Q'$, there exists some ground σ such that $F\sigma$ evaluates to true. This kind of *ordering constraint* satisfiability problems can indeed be decided for LPO [Comon 1990, Jouannaud and Okada 1991, Nieuwenhuis 1993, Nieuwenhuis and Rivero 1999] and KBO [Korovin and Voronkov 2000].

DEFINITION 5.1 The inference system of *stratified resolution* consists of two inference rules:

1. *stratified resolution rule:*

$$\frac{\underline{R} \vee C \quad \underline{\neg R'} \vee D}{(C \vee D)\sigma}\,, \qquad \begin{array}{l}\text{where } \sigma = mgu(R, R') \text{ and if } R \text{ is a } \mathcal{Q}\text{-atom,}\\ \text{then there exists a substitution } \gamma\\ \text{such that } R\sigma\gamma \succ L\sigma\gamma \text{ for all literals } L \text{ in } C.\end{array}$$

2. *positive factoring rule:*

$$\frac{\underline{Q} \vee \underline{Q'} \vee C}{(Q \vee C)\sigma}\,, \qquad \begin{array}{l}\text{where } \sigma = mgu(Q, Q') \text{ and there exists a}\\ \text{substitution } \gamma \text{ such that}\\ Q\sigma\gamma \succeq L\sigma\gamma \text{ for all literals } L \text{ in } C.\end{array}$$

Note that, in inferences involving some positive $\mathcal{Q}$-literal whose head symbol occurs as well in C, the ordering restrictions may help detecting some unnecessary inferences, but that the selection function can already filter out incompatibilities with the other ordering relations on atoms, because these relations are already fully determined by the quasi-ordering $\succeq_{pred}$ on their head symbols. Together with their use for finding a minimal subset of positive $\mathcal{Q}$-atoms for selection, this is one of the applications of the aforementioned procedures for checking ordering constraint satisfiability. Other applications arise in the context of proving the redundancy of clauses and inferences.

As in the previous section for the ground case, the following theorem is a consequence of more general results that are given below, in this case of its version with redundancy, Theorem 5.5, combined with Lemma 5.4.

THEOREM 5.2 *Stratified resolution is refutation complete for stratified sets of clauses.*

5.1 Redundancy

We now adapt to general clauses the machinery developed in Subsection 4.1 for redundancy in the ground case.

DEFINITION 5.3 Let S be a set of clauses and let $>$ be any ordering on ground clauses.

1. A *ground instance* of a stratified resolution inference

$$\frac{\underline{R} \vee C \qquad \underline{\neg R'} \vee D}{(C \vee D)\sigma}\,, \qquad \text{where } \sigma = mgu(R, R'),$$

   is any ground stratified resolution inference of the form

$$\frac{(\underline{R} \vee C)\sigma\theta \quad (\underline{\neg R'} \vee D)\sigma\theta}{(C \vee D)\sigma\theta}\,.$$

2. A *ground instance* of a positive factoring inference

$$\frac{\underline{Q} \vee \underline{Q'} \vee C}{(Q \vee C)\sigma}\,, \qquad \text{where } \sigma = mgu(Q, Q'),$$

   is any ground positive factoring inference of the form

$$\frac{(\underline{Q} \vee \underline{Q'} \vee C)\sigma\theta}{(Q \vee C)\sigma\theta}\,.$$

3. A nonground inference (by stratified resolution or positive factoring) is *redundant w.r.t. S and $>$* if all its ground instances are redundant w.r.t. $gnd(S)$ and $>$.

4. S is *saturated* iff there exists some stratified selection function for which $gnd(S)$ is saturated.

As in the ground case, in practice one can usually only use sufficient conditions showing saturatedness of a set S. In particular, we have the following.

THEOREM 5.4 *A set of clauses S is saturated if, for all inferences with premises in S, either the inference is redundant w.r.t. S and $\succ^{mul}$, or else its conclusion belongs to S.*

PROOF. Consider a selection function for $gnd(S)$ that is compatible with the one for S in the following sense: if $\underline{L} \vee C$ is in $gnd(S)$, then there is some clause $\underline{L'} \vee C'$ in S such that $L'\theta = L$ and $C'\theta = C$ for some θ. It is not difficult to argue by Zorn's lemma that such a selection function always exists.

We prove that $gnd(S)$ is saturated for any such a selection function. To this end, we take any inference π with premises in $gnd(S)$ such that the rightmost premise of π is larger w.r.t. $\succ^{mul}_{gnd(S)}$ than the conclusion, and prove that then π is redundant w.r.t. $gnd(S)$ and $\succ^{mul}_{gnd(S)}$. We consider only resolution inferences; the case of factoring is analogous. Let π be the following inference by ground stratified resolution:

$$\frac{\underline{R\theta} \vee C\theta \quad \underline{\neg R'\theta} \vee D\theta}{C\theta \vee D\theta} \tag{3}$$

By compatibility of the selection function, then indeed there exist clauses $\underline{R} \vee C$ and $\underline{\neg R'} \vee D$ (not sharing any variables) in S. Since $R\theta = R'\theta$, the atoms R and R' are unifiable with $\sigma = mgu(R, R')$. The following

$$\frac{\underline{R} \vee C \quad \underline{\neg R'} \vee D}{C\sigma \vee D\sigma} \tag{4}$$

is a valid inference by stratified resolution, because, since (3) is an inference by ground stratified resolution, if R is a $\mathcal{Q}$-atom we have $R\theta \succ L\theta$ for every literal L in C and therefore also $R\sigma\gamma \succ L\sigma\gamma$ for every literal L in C, where γ is the substitution such that $\sigma\gamma = \theta$.

Since (4) is an inference with both premises in S, either it is redundant w.r.t. S and $\succ^{mul}$, or else its conclusion belongs to S. In the former case (3) is redundant w.r.t. $gnd(S)$ and $\succ^{mul}$, because it is a ground instance of (4), and hence (3) is also redundant w.r.t. $gnd(S)$ and $\succ^{mul}_{gnd(S)}$, because $\succ^{mul}_{gnd(S)}$ extends $\succ^{mul}$. In the latter case, i.e., when $C\sigma \vee D\sigma$ is in S, we have that $C\theta \vee D\theta$ is in $gnd(S)$. Then $\neg R'\theta \vee D\theta \succ^{mul}_{gnd(S)} C\theta \vee D\theta$ implies that $C\theta \vee D\theta$ follows from a smaller w.r.t. $\succ^{mul}_{gnd(S)}$ clause in $gnd(S)$ than the maximal premise, namely from itself, and hence (4) is redundant w.r.t. $gnd(S)$ and $\succ^{mul}_{gnd(S)}$. $\square$

The following theorem is an immediate consequence of Theorem 4.8, since, by definition, a set S is saturated if so is $gnd(S)$.

THEOREM 5.5 *Let S be a saturated stratified set of clauses. Then $\square \in S$ whenever S is unsatisfiable.*

We now consider stratified resolution derivations as defined in the previous section, but applied to nonground clause sets and the nonground inference system. The only new notion that is needed is a notion of redundant clause in the nonground case:

DEFINITION 5.6 Let C be a clause, let $>$ be an ordering on ground clauses, and let S be a set of clauses. Then C is *redundant w.r.t. S and $>$* if all ground instances of C are redundant w.r.t. $gnd(S)$ and $>$.

THEOREM 5.7 *Let $S_0, S_1, \ldots$ be a fair stratified resolution derivation. Then $\square \in S_j$ for some j if, and only if, S_0 is unsatisfiable.*

PROOF. The proof proceeds like the one of Theorem 4.11, with an additional lifting argument similar to the one used in the proof of Lemma 5.4. Again each clause occurring in the derivation is a logical consequence of S_0, hence $\square \in S_j$ implies that S_0 is unsatisfiable, and again as in Theorem 4.11, for the "if" direction, it suffices to prove that the set S of persistent clauses is saturated. For proving that $gnd(S)$ is saturated, we consider again a selection function for $gnd(S)$ that is compatible with the one for S as in Lemma 5.4.

Consider any inference π with premises in $gnd(S)$. We have to show that either (i) the rightmost premise of π is not larger w.r.t. $\succ^{mul}_{gnd(S)}$ than the conclusion of π or (ii) the inference is redundant w.r.t. $gnd(S)$ and $\succ^{mul}_{gnd(S)}$. We assume that (i) does not hold, and show (ii). As in Lemma 5.4, π is an instance of some nonground inference π'. Since the derivation is fair, there exists some S_j such that either π' is redundant w.r.t. S_j and $\succ^{mul}$, or else the conclusion of π' belongs to S_j. But in both cases the conclusion of π follows from clauses in $gnd(S_j)$ smaller w.r.t. $\succ^{mul}_{gndS}$ than the rightmost premise of π. Each one of these smaller clauses in $gnd(S_j)$ follows from clauses in $gnd(S)$ that are again smaller or equal w.r.t. $\succ^{mul}$ (and hence w.r.t. $\succ^{mul}_{gnd(S)}$). Therefore, the conclusion of π follows from clauses in $gnd(S)$ that are smaller w.r.t. $\succ^{mul}_{gnd(S)}$ than the maximal premise of π, i.e., π is redundant w.r.t. $gnd(S)$ and $\succ^{mul}_{gnd(S)}$. $\square$

All the observations about deletion of unfolded clauses of Subsection 4.2 of course also apply to the nonground case handled in this section.

5.2 Redundancy and more refined orderings

The previous framework of redundancy is well-known to cover most, if not all, practical notions for the elimination of redundant inferences and clauses. As a simple example, consider *proper subsumption*, where a clause C is properly subsumed by a clause D if there exists some substitution σ such that $D\sigma \subset C$. It is clear that C is redundant w.r.t. any clause set containing D and can hence be removed.

But, up to now, for reasons of simplicity, we have considered only an ordering $\succ$ on ground atoms and a version of it for clauses. However, in some redundancy proofs it is convenient to consider more refined orderings. For instance, subsumption, where one requires only $D\sigma \subseteq C$, cannot be handled by the redundancy notions defined up to now; for example, the unit clause $r(a)$ is not redundant by these definitions in the presence of the unit clause $r(x)$.

This can again be solved by well-known techniques (see [Bachmair and Ganzinger 2000, Niewenhuis and Rubio 2000]), which we do not want to treat in detail here. Let us only mention one possibility: compare ground instances $A\sigma$ and $B\theta$ of atoms (or clauses) A and B by an ordering $\succ_{pair}$ on pairs defined by: $(A, \sigma) \succ_{pair} (B, \theta)$ if either $A\sigma \succ B\theta$ or else $A\sigma \equiv B\theta$ and A is an instance of B but not vice versa. Then adapt the definitions of redundancy accordingly.

6 A calculus for stratified resolution with redundancies

In this section we formulate a concrete derivation system of stratified resolution with redundancies in the nonground case. We will formulate rules on sets of clauses, using the symbol $\rightarrow$ to denote derivation steps.

Definition 6.1 (calculus $\mathcal{SRR}$) The calculus of *stratified resolution with redundancies*, denoted $\mathcal{SRR}$ consists of the following inference rules.

1. *Positive factoring* is the following inference rule:

$$S \rightarrow S \cup \{C\}$$

such that C is obtained from a clause in S by the positive factoring rule for clauses.

2. *Stratified resolution* is the following inference rule:

$$S \rightarrow S \cup \{C\}$$

such that C is obtained from two clauses in S by the stratified resolution rule for clauses upon a $\mathcal{Q}$-literal.

3. *Definition rewriting*. Suppose $(\underline{\neg P} \vee C) \in S$. Suppose also that $\underline{P_1} \vee D_1, \ldots, \underline{P_k} \vee D_k$ are all definition clauses in S such that P_i is unifiable with P. Then

$$S \to (S - \{\underline{\neg P} \vee C\}) \cup \{(C \vee D_1)\theta_1, \ldots, (C \vee D_n)\theta_n\},$$

where each θ_i is a most general unifier of P and P_i, is an inference by *definition rewriting*.

4. *Subsumption* and *tautology deletion* defined as usual.

Note that a derivation of definition rewriting combines *definition unfolding* of [Degtyarev and Voronkov 2000] with deletion of unfolded clauses. The completeness of this derivation system w.r.t. fair derivations follows from our results.

7 How to select a stratification

Example 3.4 shows that a set of clauses may admit several different stratifications. How can we choose a "good" stratification? When we select a stratification for a given set of clauses S, we should first find a set of predicates $\mathcal{P}$ such that S is Horn w.r.t. $\mathcal{P}$, and then select an quasi-ordering $\succeq$.

Suppose that $\mathcal{P}$ is already chosen so that S is Horn w.r.t. $\mathcal{P}$. Then we can always use the stratification in which all $\mathcal{P}$-literals are strictly greater than all $\mathcal{Q}$-literals. Unfortunately, this stratification may be not good enough, since it gives us too little choice for selecting positive $\mathcal{Q}$-literals. Let us illustrate this for clauses of Example 1.2. Assume that $\mathcal{P}$ is $\{split\}$. We can use the precedence relation

$$split \succ_{pred} deduction \succ_{pred} conference\_list.$$

This stratification does not allow us to select the literal $conference\_list(x)$ in

$$\neg split(x, y, z) \vee conference\_list(x),$$

while intuitively it should be the right selection.

This observation shows that for a given $\mathcal{P}$ it can be better to use precedence relations in which $\mathcal{Q}$-literals are as large as possible. Then we will have more options for selecting positive $\mathcal{Q}$-literals in clauses. In Example 1.2, such a more flexible stratification is based on the precedence relation

$$conference\_list \succ_{pred} split \succ_{pred} deduction.$$

In general, there is a tradeoff between the size of $\mathcal{P}$ and the flexibility of literal selection. The larger $\mathcal{P}$ is, the less choice we have for selecting positive $\mathcal{Q}$-literals.

We are planning experiments with the choice of stratification using the theorem prover Vampire [Riazanov and Voronkov 1999].

8 Conclusion

We believe that the SLD-resolution-like definition unfolding approach of stratified resolution will allow theorem provers to make the search process more goal-oriented for large classes of problems with the typical admissible kind of (recursive) definitions that can be stratified. Moreover, we believe that in practice it will be possible to find adequate stratifications automatically and inexpensively in practice. An implementation of all results exposed in this article is currently being developed inside Vampire, and we hope to be able soon to give statistical evidence of this feeling.

Finally, we now shortly mention some open problems associated with stratified resolution.

1. The standard semantics of stratified logic programs are based on nonmonotonic reasoning. Stratified resolution makes one think of a logic that combines nonmonotonic reasoning with monotonic resolution-based reasoning. Such a logic, its semantics and ways of automated reasoning in it, could be investigated. Hence it might be interesting to investigate a combination of stratified resolution with nonmonotonic logics.
2. Is there any powerful generalization of stratified resolution for logic with equality?
3. Stratified resolution is different from ordered resolution with negative selection in that it allows one to select heads of clauses, even when they are not strictly maximal in their clauses. Therefore, it may be interesting to see if stratified resolution can lead to new decision procedures for decidable fragments of predicate calculus.

Acknowledgments

The idea of this work appeared due to discussions of the third author during CADE-16 with Harald Ganzinger and Tanel Tammet. Harald Ganzinger explained that the SATURATE system tries to use an ordering under which the heads of "definition-like" clauses are the greatest in their clauses, and hence can be selected. This allows one to use definitions in the way they are often used in mathematics. Tanel Tammet explained that he tried to implement in his prover GANDALF "rewriting on the clause level", again for clauses that look like definitions.

References

APT K. [1990], Logic programming, *in* J. V. Leeuwen, ed., 'Handbook of Theoretical Computer Science', Vol. B: Formal Methods and Semantics, Elsevier Science, Amsterdam, chapter 10, pp. 493–574.

APT K. AND BLAIR H. [1988], Towards a theory of declarative knowledge, *in* J. Minker, ed., 'Foundations of Deductive Databases and Logic Programming', Morgan Kaufmann, pp. 89–148.

BACHMAIR L. AND GANZINGER H. [2000], Resolution theorem proving, *in* A. Robinson and A. Voronkov, eds, 'Handbook of Automated Reasoning', Elsevier Science and MIT Press. To appear.

COMON H. [1990], 'Solving symbolic ordering constraints', *International Journal of Foundations of Computer Science* **1**(4), 387–411.

DE NIVELLE H. [1996], Ordering Refinements of Resolution, PhD thesis, Technische Universiteit Delft.

DEGTYAREV A. AND VORONKOV A. [2000], Stratified resolution, *in* D. McAllester, ed., '17th International Conference on Automated Deduction (CADE-17)', Vol. 1831 of *Lecture Notes in Artificial Intelligence*, Springer Verlag, Pittsburgh, pp. 365–384.

GELDER A. V. [1988], Negation as failure using tight derivations for general logic programs, *in* J. Minker, ed., 'Deductive Databases and Logic Programming', Morgan Kaufmann, pp. 149–177.

JOUANNAUD J.-P. AND OKADA M. [1991], Satisfiability of systems of ordinal notations with the subterm property is decidable, *in* J. Albert, B. Monien and M. Rodríguez-Artalejo, eds, 'Automata, Languages and Programming, 18th International Colloquium, ICALP'91', Vol. 510 of *Lecture Notes in Computer Science*, Springer Verlag, Madrid, Spain, pp. 455–468.

KOROVIN K. AND VORONKOV A. [2000], A decision procedure for the existential theory of term algebras with the Knuth-Bendix ordering, *in* 'Proc. 15th Annual IEEE Symp. on Logic in Computer Science', Santa Barbara, California, pp. 291–302.

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

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

NIEUWENHUIS R. [1993], 'Simple LPO constraint solving methods', *Information Processing Letters* **47**, 65–69.

NIEUWENHUIS R. AND RIVERO J. [1999], Solved forms for path ordering constraints, *in* 'In Proc. 10th International Conference on Rewriting Techniques and Applications (RTA)', Vol. 1631 of *Lecture Notes in Computer Science*, Trento, Italy, pp. 1–15.

NIEWENHUIS R. AND RUBIO A. [2000], Paramodulation-based theorem proving, *in* A. Robinson and A. Voronkov, eds, 'Handbook of Automated Reasoning', Elsevier Science and MIT Press. To appear.

PRZYMUSINSKI H. [1988], On the declarative semantics of deductive databases and logic programs, *in* J. Minker, ed., 'Foundations of Deductive Databases and Logic Programming', Morgan Kaufmann, pp. 193–216.

RIAZANOV A. AND VORONKOV A. [1999], Vampire, *in* H. Ganzinger, ed., 'Automated Deduction—CADE-16. 16th International Conference on Automated Deduction', Vol. 1632 of *Lecture Notes in Artificial Intelligence*, Trento, Italy, pp. 292–296.