Resolution Games and Non-Liftable Resolution Orderings

Hans de Nivelle,
Department of Mathematics and Computer Science,
Delft University of Technology,
Julianalaan 132, 2628 BL, the Netherlands,
email: nivelle@cs.tudelft.nl

Abstract

We prove the completeness of the combination of ordered resolution and factoring for a large class of non-liftable orderings, without the need for any additional rules like saturation. This is possible because of a new proof method wich avoids making use of the standard ordered lifting theorem. This proof method is based on resolution games.

1 Introduction

Resolution was introduced in ([Robins65]) and is still among the most successful methods for automated theorem proving in first order logic. (See [ChangLee73]). Although resolution is efficient, it is not efficient enough. Therefore so called *refinements* of resolution have been designed, which can improve efficiency quite a lot, without losing completeness. In this paper we will consider *ordering refinements*. Ordering refinements are a restriction of the resolution rule. With refinements two types of improvement can be gained: First resolution refinements simply improve efficiency, which means that less memory will be used, and less time will be spent on finding a proof if it exists. Second it can be shown that certain resolution refinements are *terminating* on certain clause sets for which unrestricted resolution would be non-terminating. Thus it is possible to obtain *decision procedures* with resolution. This approach was initiated by ([Joy76]) and ([Zam72]). ([FLTZ93]) contains an overview of the results reached in this field. The general strategy for proving the completeness of an ordering refinement is as follows: **(1)** Prove the completeness of the refinement for the ground level. **(2)** Then show that a refutation of a certain set of ground clauses can be *lifted* to the non-ground level. For the first part it has been shown that resolution with every ordering on ground literals is complete. For the second part, the ordering must have a special property which is called *liftability*: $A \prec B \Rightarrow A\Theta \prec B\Theta$. This

property is problematic because any ordering that satisfies this property must leave many literals uncompared. For example literals $p(X)$ and $p(s(Y))$ cannot be compared. Suppose that $p(X) \prec p(s(Y))$. Then because of liftability, $\prec$ is preserved by both $\{X := s(s(0)), Y := 0\}$, and $\{X := s(0), Y := s(0)\}$. However this results in $p(s(s(0))) \prec p(s(0))$ and $p(s(0)) \prec p(s(s(0)))$. This contradicts the fact that $\prec$ is an ordering. In the same way $p(s(Y)) \prec p(X)$ is impossible.
From the efficiency point of view it is desirable to compare as many literals as possible, because then as little as possible resolution inferences will be made.
It is also desirable from the decision point of view to be able to drop this liftability property, because certain non-liftable orderings have been proven terminating for certain clause sets, but it was not known whether or not these orderings were complete. (See [FLTZ93]). We can positively answer this question here.
It is for these reasons that we will study resolution with non-liftable orderings here. We will prove two completeness theorems for two types of non-liftable orderings. For these proofs we will make use of a device called *resolution game*. The resolution game can be seen as ordered resolution, with which a certain counterplayer can change the ordering at certain moments. We begin by repeating some basic definitions.

Definition 1.1 An *order* is a relation which satisfies the following properties: **O1** For no d is it the case that $d < d$. **O2** For each d_1, d_2 and d_3: if $d_1 < d_2$, $d_2 < d_3$, then $d_1 < d_3$. An order $<$ is *total* if **O3** whenever $d_1 \neq d_2$, then either $d_1 < d_2$ or $d_2 < d_1$.
$<$ is *well-founded* if there is no infinite sequence $d_0, d_1, d_2, \ldots$, such that $d_0 > d_1 > d_2 > \cdots > d_i > \cdots$.
A total, well-founded relation is called a *well-order*.

Definition 1.2 Let F be a finite set of function symbols with arities attached to them, V be a countably infinite set of variables, and let P be a finite set of predicate symbols with arities attached to them. We define *terms* as follows: **(1)** Every variable, or function in F with arity 0 is a term. **(2)** If f is an element of F, with arity n, and $t_1, \ldots, t_n$ are terms, then $f(t_1, \ldots, t_n)$ is a term. There are no other terms than defined by these rules.

If p is a predicate symbol, with arity n, and $t_1, \ldots, t_n$ are terms, then $p(t_1, \ldots, t_n)$ is an *atom* . A *literal* is an atom $p(t_1, \ldots, t_n)$, or its negation $\neg\ p(t_1, \ldots, t_n)$. The *complexity* of a term $\#t$ is defined from $\#v = 1$, for a variabele v, $\#f = 2$, for a 0-ary function symbol $f \in F$, and $f(t_1, \ldots, t_n) = 2 + \#t_1 + \cdots \#t_n$, for an n-ary function symbol $f \in F$.

Definition 1.3 A *substitution* Θ is a finite set of the form $\{v_1 := t_1, \ldots, v_n := t_n\}$, where each $v_i \in V$ and each t_i is a term. The *effect* of substitution Θ on a literal A is defined as usual: As the result of replacing simultaneously all v_i in A by t_i. Because of this it must be the case that for no $i \neq j$, we have $v_i = v_j$. Otherwise the effect on literals containing $v_i = v_j$ would be undefined.

A literal A is an *instance* of a literal B if A can be obtained from B by a substitution. We call A a *renaming of* B if A is an instance of B and B is an instance of A. In that case we also call A and B *equivalent*. We call A a *strict instance* of B if A is an instance of B and A and B are non-equivalent.

If A and B are literals and Θ is a substitution, such that $A\Theta = B\Theta$ then both Θ and $A\Theta$ are called a *unifier* of A and B.

If Θ is a unifier of A and B, then Θ is called a *most general unifier* if for every unifier Σ of A and B, and every literal C, it is the case that $C\Sigma$ is an instance of $C\Theta$.

If A is an instance of B and $\Theta = \{v_1 := t_1, \ldots, v_n := t_n\}$ is a $\subseteq$-minimal substitution such that $A = B\Theta$, then we define the *complexity of the instantiation* as $\#t_1 + \cdots + \#t_n$.

It has been proven in [Robins65] that there exists an algorithm that has as input two atoms (or literals), computes a most general unifier if they are unifiable, and reports failure otherwise.

Definition 1.4 A *clause* is a finite set of literals. A clause $\{A_1, \ldots, A_p\}$ should be read as the first order formula $\forall \overline{x}(A_1 \vee \cdots \vee A_p)$. Here $\overline{x}$ are the variables that occur in the clause.

We call a clause *decomposed* if all literals in it have exactly the same variables. An *L-ordering* $\sqsubset$ is an ordering on literals. If $\sqsubset$ is an L-order, then a literal L is *maximal* in a clause c if **(1)** $L \in c$, and **(2)** for no $L' \in c$, we have $L \sqsubset L'$.

Note that, because $\sqsubset$ is an order, and clauses are finite, every non-empty clause has at least one maximal element.
Resolution is a *refutation method*. If one wants to try to prove a formula one has to try to refute its negation.

Definition 1.5 Ordered resolution We define the resolution rule: Let c_1 and c_2 be clauses, such that **(1)** c_1 and c_2 can be written as $c_1 = \{A_1\} \cup r_1$, and $c_2 = \{\neg A_2\} \cup r_2$, **(2)** A_1 is $\sqsubset$-maximal in c_1, and $\neg A_2$ is $\sqsubset$-maximal in c_2, and **(3)** A_1 and A_2 are unifiable with mgu Θ. Then $r_1\Theta \cup r_2\Theta$ is an *ordered resolvent* of c_1 and c_2. We write $c_1, c_2 \vdash r_1\Theta \cup r_2\Theta$.

Ordered factoring Let c be a clause containing 2 literals A_1 and A_2, such that **(1)** A_1 and A_2 are unifiable with mgu Θ, and **(2)** A_1 is $\sqsubset$-maximal in c. Then $c\Theta$ is an ordered factor of c. Notation $c \vdash_f c\Theta$.

We have not defined unrefined resolution. Unrefined resolution can be obtained by dropping the ordering conditions in definition 1.5.

Definition 1.6 We call $\sqsubset$ *liftable* if $A \sqsubset B \Rightarrow A\Theta \sqsubset B\Theta$.

This property ensures that if a literal $A_i\Theta$ is maximal in a clause $\{A_1\Theta, \ldots, A_p\Theta\}$, that then its uninstantiated counterpart A_i in $\{A_1, \ldots, A_p\}$ is also maximal. This makes lifting possible. The next theorem is the standard ordered resolution theorem.

Theorem 1.7 Ordered resolution with ordered factoring is complete, for any liftable L-order.

L-orders are a slight generalization of the more well-known A-orders. An A-order is an order on atoms, which is extended to literals by the rule $A \sqsubset B \Rightarrow A \sqsubset \neg B, \neg A \sqsubset B, \neg A \sqsubset \neg B$. Although every extension of an A-order is an L-order, the converse is not true. For example $P \sqsubset Q \sqsubset \neg Q \sqsubset \neg P$ is an L-order, but not the extension of an A-order. It is known that A-ordered resolution and factoring is complete since ([KH69]).

2 Non-Liftable Orderings

We will now give the two completeness theorems for non-liftable orderings. For the proof we develope the resolution games in the next section. After that we prove the two completeness theorems in Section 5.

Theorem 2.1 Let $\sqsubset$ be an L-order, such that

REN If $A \sqsubset B$, then for all renamings $A\Theta_1$ of A, and $B\Theta_2$ of B, we must have $A\Theta_1 \sqsubset B\Theta_2$,

SUBST For every A and strict instance $A\Theta$ of A it must be that $A\Theta \sqsubset A$.

Then the combination of $\sqsubset$-ordered resolution and factoring is complete.

Theorem 2.1 implies the completeness of resolution with any relation that is included in an order satisfying the conditions. An example is the ordering defined by $L_1 \sqsubset L_2$ iff $\#L_1 > \#L_2$. Another possibility is an alfabetic, lexicographic ordering on term structure.

Theorem 2.2 Let $\sqsubset$ be an order, such that

REN if A and B contain exactly the same variables, and $A \sqsubset B$, then for all substitutions Θ_1 and Θ_2, such that **(1)** $A\Theta_1$ is a renaming of A, **(2)** $B\Theta_2$ is a renaming of B, **(3)** $A\Theta_1$ and $B\Theta_2$ have exactly the same variables, we have $A\Theta_1 \sqsubset B\Theta_2$.

Then $\sqsubset$-ordered resolution with factoring is complete for every set of decomposed clauses.

It is impossible that $p(X,Y) \sqsubset q(X,Y)$ and $q(Y,X) \sqsubset p(X,Y)$. This would imply $p(X,Y) \sqsubset p(Y,X)$, which would imply $p(X,Y) \sqsubset p(X,Y)$.

The $<_v$ order, together with the $E+'$-class, defined in ([FLTZ93]), pp. 82, satisfies the conditions, mentioned here. There is no place for details here, but the check is easy.

3 Resolution Games

In this section we define resolution games and give a completeness result for resolution games. The proof is given in the next section. We need a precise control over the factoring rule. Therefore it is needed to define clauses as multisets instead of ordinary sets. So we define:

Definition 3.1 A *multiset* is a set, which is able to distinguish how often an element occurs in it. We write $[A_1, \dots, A_p]$ for the multiset containing $A_1, \dots, A_p$. Unlike in the set $\{A_1, \dots, A_p\}$ it is meaningful to repeat elements in the list. The *union* of 2 multisets $S_1 \cup S_2$ is obtained by summing the number of occurrences for each element. The *difference set* of 2 multisets $S_1 \backslash S_2$ is obtained by subtracting for each element, the number of occurrences in S_2 from the number of occurrences in S_1. If this results in a negative number then the number of occurrences is put to 0.

Definition 3.2 A (binary) *resolution game* is an ordered triple, $\mathcal{G} = (P, \mathcal{A}, \prec)$, where

- P is a set of propositional symbols. We define a *literal* of $\mathcal{G}$ as a propositional symbol p or its negation $\neg\, p$.
- $\mathcal{A}$ is a set of attributes,
- $\prec$ is an order on $\mathcal{L} \times \mathcal{A}$, where $\mathcal{L}$ is the set of literals. It must be the case that $\prec$ is well-founded on $\mathcal{L} \times \mathcal{A}$.

An *indexed literal* is a pair $L : a$ consisting of a literal L and an attribute $a \in \mathcal{A}$. A *clause* of $\mathcal{G}$ is a finite multiset of indexed literals of $\mathcal{G}$.

The meaning of a clause is the disjunction of its literals. So the clause $[a_1 : A_1, \dots, a_p : A_p]$ has as meaning $a_1 \vee \cdots \vee a_p$. Accordingly we call a set of clauses satisfiable if the set of its meanings is satisfiable. We define:

Definition 3.3

Resolution Let c_1 and c_2 be two clauses, such that **(1)** c_1 can be written as $c_1 = [r : R_1] \cup [a_1 : A_1, \dots, a_p : A_p]$, and c_2 can be written as $c_2 = [\neg\, r : R_2] \cup [b_1 : B_1, \dots, b_q : B_q]$, **(2)** $r : R_1$ is $\prec$-maximal in c_1, and $\neg\, r : R_2$ is $\prec$-maximal in c_2. Then $[a_1 : A_1, \dots, a_p : A_p] \cup [b_1 : B_1, \dots, b_q : B_q]$ is a resolvent of c_1 and c_2.

Factoring Let $c = [a_1 : A_1, \dots, a_p : A_p]$ be a clause, such that: **(1)** $a_1 : A_1$ is maximal in c, **(2)** $a_1 = a_i$, for an $i > 1$. Then $c \backslash [a_i : A_i]$ is a factor of c.

Reduction Let $c = [a_1 : A_1, \dots, a_p : A_p]$ be a clause. A reduction of c is obtained by replacing zero, one or more $a_i : A_i$ by an $a_i : A_i'$, such that $a_i : A_i' \prec a_i : A_i$. It is also possible to delete literals in the clause. (Note that there is no maximality restriction here).

We can now define how the game is played.

Definition 3.4 Let C be a finite set of clauses of a resolution game $\mathcal{G}$. There are two players.

The opponent The opponent will try to derive the empty clause by computing factors and resolvents.

The defender The defender will try to prevent this by replacing newly derived clauses by reductions.

There are two sets G and N. The set G contains all the derived clauses, and N contains the clauses of the last generation. The game starts with $G = \emptyset$, and $N = C$. Then:

1. The defender can replace any clause in N by a reduction. So he can make 0, 1 or any finite number of replacements. When the defender is finished N is added to G. N is emptied.

2. Now the opponent can compute any ordered resolvent, or ordered factor of clauses in G. The result is put in N. He can derive as many clauses as he wants in one turn, but he cannot use the new clauses because they are in N. When he is finished the defender is on turn again.

The game ends when the opponent succeeds in deriving the empty clause. In that case the opponent is the winner. If the defender succeeds in preventing this, the defender is the winner. Unfortunately for him, he will not enjoy his victory at a finite time, because in this case the game may last forever.

We have defined the resolution game in such a way that the defender can only affect newly derived clauses. We could also have defined the resolution game in such a way that the defender is allowed to reduce any clause. In that case Theorem 3.5 still holds.
The resolution game is different from lock or indexed resolution [Boyer71], because in lock resolution the resolvent inherits the indices from the parent clause without any changes. We have the following theorem:

Theorem 3.5 Let C be a set of clauses of a resolution game $\mathcal{G}$. **(1)** If C is unsatisfiable, then the opponent of the resolution game can play in such a way that he is guaranteed to derive the empty clause at a finite moment. **(2)** If C is satisfiable then the defender can play in such a way that the opponent will not derive the empty clause.

We call the first part of the theorem *completeness*, and the second part *soundness*. The proof of the soundness is not difficult. All the actions of the opponent are semantically sound. The defender can play in such a manner that his actions are sound, by never deleting a literal. This guarantees that the empty clause will not be derived if C is satisfiable. The proof of the completeness is more difficult. We give the main part of it in the next section. Here we only show that it is sufficient to consider resolution games $\mathcal{G} = (P, \mathcal{A}, \prec)$, in which $\prec$ is total. We have the following lemma.

Lemma 3.6 Every well-founded order is contained in a well-order.

So for every resolution game $\mathcal{G}$ it is possible to obtain a resolution game $\mathcal{G}'$ by replacing $\prec$ by a well-order $\prec'$. Then we have:

COPY1 Every resolvent, or factor that can be computed with $\mathcal{G}'$, can also be computed with $\mathcal{G}$. This is because a literal, that is maximal w.r.t to $\prec'$ will certainly be maximal w.r.t. $\prec$.

COPY2 Every reduction that can be made with $\mathcal{G}$ is also a reduction with $\mathcal{G}'$.

We will show that the completeness of game $\mathcal{G}'$, implies the completeness of $\mathcal{G}$. It is for this reason that it is sufficient to consider games in which the order is total. An opponent of a set of clauses C playing $\mathcal{G}$ can simultaneously play a game using game $\mathcal{G}'$ as defender. He will copy the moves from the opponent of $\mathcal{G}'$ to $\mathcal{G}$, and copy the moves from the defender of $\mathcal{G}$ to $\mathcal{G}'$. This goes as follows: The opponent of a set of clauses C with game $\mathcal{G}$ starts a simultaneous game as defender of C using game $\mathcal{G}'$. Then he proceeds as follows:

1. He waits for the defender on game $\mathcal{G}$ to make his reductions.
2. After this he can imitate the reductions made by the defender on $\mathcal{G}$ onto $\mathcal{G}'$. This is possible because of COPY2.
3. Then he waits for the opponent of $\mathcal{G}'$ to compute his factors and resolvents.
4. When the opponent of game $\mathcal{G}'$ is finished he imitates his moves on $\mathcal{G}$. This is possible because of COPY1. After this he continues at 1.

Because the opponent of $\mathcal{G}'$ will derive the empty clause, if the initial clause set is unsatisfiable, the opponent of $\mathcal{G}$ will derive the empty clause, and win the resolution game. So it is sufficient to prove the completeness of resolution games for those resolution games, in which $\prec$ is a well-order. We will do this in the next section. We will end with an example:

Example 3.7 Let $\prec$ be defined from:

$\neg\, c:0 \prec$ $\quad b:0 \prec$ $\quad \neg\, a:0 \prec$ $\quad \neg\, a:1 \prec$ $\quad b:1 \prec$ $\quad c:0 \prec$ $\quad \neg\, c:1 \prec$
$a:0 \prec$ $\quad c:1 \prec$ $\quad \neg\, c:2 \prec$ $\quad \neg\, b:0 \prec$ $\quad \neg\, a:2 \prec$ $\quad \neg\, b:1 \prec$ $\quad b:2 \prec$
$c:2 \prec$ $\quad \neg\, b:2 \prec$ $\quad a:1 \prec$ $\quad a:2.$

Let C be the following unsatisfiable set of clauses:

$[b:2, c:2, a:2]$ $\quad [c:2, \neg\, b:2]$ $\quad [\neg\, c:2]$ $\quad [\neg\, a:2, b:2].$

The clauses are sorted according to $\prec$. So each last literal is the selected literal. If the defender doesn't make any reductions then the resolvent $[\neg\ a : 2, c : 2]$ is possible. This clause can be reduced to for example $[\neg\ a : 0, c : 0]$, $[c : 1, \neg\ a : 2]$, or $[\neg\ a : 0, c : 2]$. The defender can also replace the initial clause $[c : 2, \neg\ b : 2]$ by $[\neg\ b : 1, c : 2]$. In that case the only possible resolvent is $[\neg\ b : 1]$. Whatever reductions the defender makes, the emtpy clause can always be derived.

4 Completeness of Resolution Games

In this section we give the completeness proof of the resolution game. For this proof we need the following notion:

Definition 4.1 Let $\overline{C}$ be a set of clauses. We call $\overline{C}$ closed iff

1. For every $c_1, c_2 \in \overline{C}$, such that c_1 and c_2 have a resolvent d, there is a reduction d' of d in $\overline{C}$.

2. For every $c \in \overline{C}$, such that c has a factor d, there is a reduction d' of d in $\overline{C}$.

$\overline{C}$ is a *closure* of a clause set C if $\overline{C}$ contains a reduction of every $c \in C$.

We will prove completeness of resolution games by showing that every closed set that does not contain the empty clause, is satisfiable. This implies completeness. Suppose that this holds, while resolution games are not complete. There is a clause set C, of a resolution game $\mathcal{G}$, such that C is unsatisfiable, and whatever the opponent does, the defender can block derivation of the empty clause. Then, when the opponent produces all possible clauses in each move, the conjunction of the successive generations $\overline{C} = \bigcup_{i>0} G_i$ is a closure of C. By assumption this set does not contain the empty clause. Then $\overline{C}$ must be satisfiable, and this implies that C is satifiable. This is a contradiction.
We use an adaptation of a proof in [Bezem90], of the completeness of A-ordered hyperresolution. The proof is probably a bit dissapointing, because it does not use the game-structure, but it is with less technicality than the proof in ([Nivelle94b]). The proof in ([Nivelle94b]) is based on games. We adapt the proof in two steps for the clarity of the presentation. We first give the proof for the case in which the defender never makes a reduction. In that case we have proven the completeness of a variant of lock resolution. After that we make some more adaptations to obtain the completeness of the full resolution game. We will show that every closed set of clauses has a formal model, and that this implies that every closed set of clauses has a model.

Definition 4.2 Let C be a set of clauses of a resolution game $\mathcal{G}$. We define a *formal model* M as a set of indexed literals, which **(1)** does not contain a complementary pair, $\neg\ a : i_1$, and $a : i_2$, **(2)** and which contains an indexed literal of every clause.

We have the following simple lemma:

Lemma 4.3 If a set of clauses C has a formal model M, then it has a model.

This can be seen by taking the interpretation I, defined by: $I(A) = t$ iff an $A : i$ occurs in M, for each atom A.

4.1 Completeness of Restricted Resolution Games

If we consider games in which the defender never makes a reduction we have that $d = d'$, in both cases of Definition 4.1.

Definition 4.4 Let $\overline{C}$ be a closed set of clauses. We define an *intersection set* of $\overline{C}$, as a set of indexed literals I, s.t. I contains a literal of every $c \in \overline{C}$.

We will construct an intersection set I, s.t.

MAXUNIQUE for every $A : a \in I$, there is a clause $c \in \overline{C}$, such that $A : a$ is maximal in c, $A : a$ is not repeated in c, and there is no other indexed literal of c in I.

It is the case that if a certain set I is an intersection set of a set of clauses $\overline{C}$ and I satisfies MAXUNIQUE, then I is a formal model of $\overline{C}$. This is seen as follows: Suppose that I contains a complementary pair $A : a_1$ and $\neg\ A : a_2$. Then there are clauses c_1 and c_2 such that $A : a_1$ is maximal in c_1 and $c_1 \backslash [A : a_1] \cap I = \emptyset$. and $\neg\ A : a_2$ is maximal in c_2 and $c_2 \backslash [\neg\ A : a_2] \cap I = \emptyset$. Now because $\overline{C}$ is closed under resolution, $\overline{C}$ contains $d = (c_1 \backslash [A : a_1]) \cup (c_2 \backslash [\neg\ A : a_2])$. Then $d \cap I = \emptyset$ and this contradicts the fact that I is an intersection set.
So what remains to show is that there exists an intersection set, satisfying MAXUNIQUE. We will construct this intersection set.

Lemma 4.5 Let $\overline{C}$ be a closed set (in which resolvents and factors are never reduced), s.t. $\emptyset \notin \overline{C}$. Then there exists an intersection set I of $\overline{C}$, that satisfies MAXUNIQUE.

Proof: Because $\prec$ is a well-order on the set of indexed literals we can use recursion. Let λ be the ordinal length of $\prec$. Let L_α be the α-th indexed literal, for $0 \leq \alpha < \lambda$. With I_α we will denote the construction of the set I up to α.
We construct the I_α as follows:

1. $I_0 = \emptyset$,
2. For any limit ordinal α, let $I_\alpha = \bigcup_{\beta < \alpha} I_\beta$.

3. For any successor ordinal α, put $I_\alpha = I_{\alpha-1}$ if $I_{\alpha-1} \cup \{L_\beta \mid \alpha \leq \beta < \lambda\}$ is an intersection set. Otherwise let $I_\alpha = I_{\alpha-1} \cup \{L_{\alpha-1}\}$. (So at stage α we decide whether or not $L_{\alpha-1}$ is added)

4. Finally put $I = I_\lambda$.

We will show that I is an intersection set satisfying MAXUNIQUE.
Suppose that I is not an intersection set. Then there is a clause $c \in \overline{C}$, such that $I \cap c = \emptyset$. Let α be the index of the maximal literal in c. So L_α is the maximal literal of c. Then at stage $\alpha + 1$ of the construction, $I_\alpha \cup \{L_{\alpha+1}, L_{\alpha+2}, \ldots\}$ is not an intersection set, and L_α would have been added to I_α. This is a contradiction.
It remains to prove that I satisfies MAXUNIQUE. Suppose I does not. Then I contains an indexed literal $A : a = L_{\alpha-1}$ such that either

1. $L_{\alpha-1}$ does not occur uniquely in a clause $c \in \overline{C}$. Then at stage α of the construction of I, $L_{\alpha-1}$ would not have been added.

2. $L_{\alpha-1}$ does occur uniquely in some clauses, but nowhere as maximal element. In that case the set $\{L_\beta \mid \alpha \leq \beta < \lambda\}$ contains all maximal elements of clauses in which $L_{\alpha-1}$ uniquely occurs, and $L_{\alpha-1}$ would not have been added at stage α.

3. $L_{\alpha-1}$ occurs uniquely and maximally in a clause c, and as maximal element, but is repeated. In that case there is a (possible iterated) factor of c in $\overline{C}$, in which $L_{\alpha-1}$ is not repeated.

End of proof

4.2 Completeness of Full Resolution Games

We will now adapt this proof to a completeness proof for full resolution games. The first problem that we encounter is that the argument below the definition of MAXUNIQUE does not work anymore, because the resolvent may be reduced. We have to replace Definition 4.4 by

Definition 4.6 Let $\overline{C}$ be a closed set of clauses. We define an intersection set of $\overline{C}$ as a set of indexed literals, s.t.

1. If $A : a_1 \in I$, then for all $A : a_2$, such that $A : a_1 \prec A : a_2$, also $A : a_2 \in I$.

2. From every clause $c \in \overline{C}$, there is an element in I.

Then we can replace property MAXUNIQUE by

MAXUNIQUE2 For every $A : a_1 \in I$, for which there is no $A : a_2 \in I$, such that $A : a_2 \prec A : a_1$, there is a clause $c \in \overline{C}$, such that $A : a_1$ is maximal in c, $A : a_1$ is not repeated in c, and there is no other indexed literal of I in c.

Now it is possible to repeat the argument below the definition of MAXUNIQUE with a few adaptations. Suppose that an intersection set I of $\overline{C}$ which satisfies MAXUNIQUE2, contains a complementary pair $A : a_1$ and $\neg\ A : a_2$. Then I contains minimal elements $A : a_1'$ and $\neg\ A : a_2'$ for which $A : a_1' \prec A : a_1$ and $\neg\ A : a_2' \prec \neg\ A : a_2$. For these literals there must be clauses c_1 and c_2, such that $A : a_1'$ is maximal in c_1, $\neg\ A : a_2'$ is maximal in c_2, and $((c_1 \backslash [A : a_1']) \cup (c_2 \backslash [\neg\ A : a_2']))\ \cap I = \emptyset$. Then $\overline{C}$ contains a reduction d' of this resolvent. It must be the case that $d' \cap I = \emptyset$, because of property 1 in Definition 4.6, and this contradicts property 2 in Definition 4.6. So it remains to show that there exists an intersection set, satisfying MAXUNIQUE2.

Lemma 4.7 Let $\overline{C}$ be a closed set of clauses, for which $\emptyset \notin \overline{C}$. There exists an intersection set I of $\overline{C}$, that satisfies MAXUNIQUE2.

Proof: Let $\overline{C}_f \subseteq \overline{C}$ be the set of clauses of $\overline{C}$ with non-repeated maximal elements, i.e., the set of clauses that does not have a factor. We use the same recursion as in the proof of Lemma 4.5. Let λ be the length of $\prec$. Let L_α be the α-th indexed literal, for $0 \leq \alpha < \lambda$. Let I_α be the construction of I up to α, (here $0 \leq \alpha \leq \lambda$) The construction goes as follows:

1. $I_0 = \emptyset$,
2. For any limit ordinal α, put $I_\alpha = \bigcup_{\beta<\alpha} I_\beta$.
3. For any successor ordinal α do
   (a) If there is a literal $A : a' \in I_{\alpha-1}$, such that $L_{\alpha-1} = A : a$ and $A : a' \prec A : a$, then $I_\alpha = I_{\alpha-1} \cup \{L_{\alpha-1}\}$.
   (b) Otherwise (if no such literal exists), then
      i. if $I_{\alpha-1} \cup \{L_\beta \mid \alpha \leq \beta < \lambda\}$ is an intersection set of $\overline{C}_f$, then $I_\alpha = I_{\alpha-1}$.
      ii. otherwise $I_\alpha = I_{\alpha-1} \cup \{L_{\alpha-1}\}$.
4. Finally we define $I = I_\lambda$.

It is not difficult to see that I is an intersection set of $\overline{C}$, because every intersection set of $\overline{C}_f$ is an intersection set of $\overline{C}$. We must show that I satisfies MAXUNIQUE2. Let $A : a_1$ be such that there is no $A : a_2 \in I$, for which $A : a_2 \prec A : a_1$ and despite this, there is no clause $c \in \overline{C}_f$, such that $A : a_1$ is maximal in c, and $A : a_1$ is the only literal of I in c. Let α be the moment at which adding of $A : a_1$ is decided, so $A : a_1 = L_{\alpha-1}$. There are the following possibilities:

1. $L_{\alpha-1}$ does not occur uniquely in a clause $c \in \overline{C}_f$. Then at stage α, $L_{\alpha-1}$ would not have been added.

2. $L_{\alpha-1}$ does occur uniquely in some clauses in $\overline{C}_f$, but nowhere as maximal element. In that case $I_{\alpha-1} \cup \{L_\beta \mid \alpha \leq \beta < \lambda\}$ is an intersection set, and $L_{\alpha-1}$ would not have been added.

3. $L_{\alpha-1}$ does occur uniquely and maximally in a clause $c \in \overline{C}_f$, but is repeated. This is impossible because of the nature of $\overline{C}_f$.

End of proof
We will give two examples demonstrating that resolution games are not complete when **(1)** the condition that $A : a' \prec A : a$ in reductions, or **(2)** the condition that $\prec$ is well-founded, is dropped. (So for example replacing $a : 1$ by $a : 2$ is a valid reduction in the first case)

Example 4.8 Define $\mathcal{G} = (P, \mathcal{A}, \prec)$ from $P = \{a, b\}$, $\mathcal{A} = \mathcal{N}$, and $l_1 : n_1 \prec l_2 : n_2$ iff $n_1 < n_2$. The clause set $C =$

$[a : 0, b : 1]$	$[\neg\, b : 0, a : 1]$	$[\neg\, a : 0, \neg\, b : 1]$	$[b : 0, \neg\, a : 1]$
$[a : 0, \neg\, a : 1]$	$[b : 0, \neg\, b : 1]$		

is closed and unsatisfiable, but does not contain the empty clause.
Now replace $\mathcal{N}$ by $\mathcal{Z}$. Then $\prec$ is not well-founded anymore. Let the initial clause set be equal to C. The defender can always reduce in such a manner that newly derived clauses are sorted in the same way as in C. Therefore he can block derivation of the empty clause.

5 Application of Resolution Games

We are now in the position to prove Theorems 2.1 and 2.2. For both theorems the strategy is the same. Each unsatisfiable clause set has a finite set C_g of ground-instances, which is unsatisfiable. From this non-satisfiable set we construct the resolution game, by taking all the ground literals in C_g. We use the attributes of the resolution game to indicate the non-ground literals by which the ground literals are represented. Then it is possible for the defender to make his moves in such a manner, that the resulting game corresponds to the behaviour of the non-liftable ordering. Because the empty clause will be derived with the game, the empty clause will be derived with the non-liftable ordering.
We begin with Theorem 2.1. Assume that a set of clauses C is unsatisfiable.
By Herbrands theorem there exists a finite set $C_g = \{\overline{c}_1, \ldots, \overline{c}_n\}$ of clauses in C, such that C_g is unsatisfiable. Let $C_{used} = \{c_1, \ldots, c_n\} \subseteq C$ be the set of clauses for which each $\overline{c}_i$ is an instance of c_i. (Here C_{used} is written with possible repetitions)
Now construct the following resolution game. Define $\mathcal{G} = (P, \mathcal{A}, \prec)$, where

- P is the set of ground atoms occurring in C_g. P is finite, because C_g is finite. (We will denote the set of literals that can be formed from elements of P as $\mathcal{L}$).

- We define $\mathcal{A}$ as the set of literals, s.t. each $L \in \mathcal{A}$
  1. has an instance in a clause of C_g, and
  2. is an instance of a literal, occurring in a clause in C_{used}.
- $\prec$ is defined as follows. $(a_1 : A_1) \prec (a_2 : A_2)$ if $A_1 \sqsubset A_2$.

We will show that $\mathcal{G}$ is a valid resolution game. For this we have to show that $\prec$ is an order on $\mathcal{L} \times \mathcal{A}$, and that $\prec$ is well-founded on $\mathcal{L} \times \mathcal{A}$. The first follows trivially from the fact that $\sqsubset$ is an order.
For the second let us define $a_1 : A_1 \equiv a_2 : A_2$ if $a_1 = a_2$, and A_1 is equivalent with A_2 (i.e. they are an instance of each other).
This is an equivalence relation with only a finite number of equivalence-classes. $\prec$ will not distinguish elements of these classes. Then, because every sequence of $\prec$ must be finite, $\prec$ is well-founded.
We will now describe how the resolution game is played.

- The resolution game starts with the following set of clauses: For every $c_i = \{A_1, \ldots, A_p\}$, the initial set C_{game} contains a clause $[A_1\Theta : A_1, \ldots, A_p\Theta : A_p]$. Here Θ is a substitution, such that $c_i\Theta = \overline{c}_i$. In his first move the defender does not affect the indices. Now we have:

  INSTANCE There exists, for each initial clause $[a_1 : A_1, \ldots, a_p : A_p]$ one substitution Θ, such that for all $a_i : A_i$, we have $A_i\Theta = a_i$.

  This property will be preserved throughout the game by the defender.

- When the opponent derives a clause $c = [a_1 : A_1, \ldots, a_p : A_p]$, by resolution, and $p : P_1$, and $\neg\ p : P_2$ are the literals resolved upon in the parent clauses, the defender reacts by replacing all A_i by $A_i\Theta$, where Θ is the mgu of $\neg\ P_1$ and P_2. After this he deletes all repeated occurrences of indexed literals in the result.

- When the opponent derives a clause $c = [a_1 : A_1, \ldots, a_p : A_p]$, by factorization, and $p : P_1$, and $p : P_2$ are the literals factored upon, then the defender reacts by replacing all A_i by $A_i\Theta$, where Θ is the mgu of P_1 and P_2. Afther this he deletes all repeated occurences of indexed literals in the result.

This is valid strategy because property INSTANCE will be preserved throughout the game by the defender. The defender will lose the resolution game. From this game a $\sqsubset$-ordered refutation of C can be extracted, by replacing each clause $[a_1 : A_1, \ldots, a_p : A_p]$ by the clause $\{A_1, \ldots, A_p\}$.
We can now prove Theorem 2.2. Let C be a unsatisfiable set of decomposed clauses, let C_{used} and C_g be defined as in the proof of Theorem 2.1.

The resolution game will be a little different. $\mathcal{G} = (P, \mathcal{A}, \prec)$. P and $\mathcal{A}$ are constructed in the same way, but $\prec$ is constructed different.

- $\prec$ is defined from: $(a_1 : A_1) \prec (a_2 : A_2)$ if one of the following holds

  1. The complexity of the instantiation (A_1 becomes a_1) is strictly less than the complexity of the instantiation (A_2 becomes a_2)
  2. The complexity of the instantiation (A_1 becomes a_1) is equal to the complexity of the instantiation (A_2 becomes a_2), A_1 and A_2 have the same number of variables. Then there exists a renaming $A_1\Theta$ of A_1, such that $A_1\Theta$ has the same variables as A_2. Then it must be the case that: $A_1\Theta \sqsubset A_2$. Note that if these conditions hold for one Θ, such that $A_1\Theta$ has the same variables as A_2, then they hold for any such Θ', because of REN in Theorem 2.2.

We will show that this is a valid resolution game. In order to show that the relation $\prec$ is a well-founded order, it is sufficient to show that the relations mentioned under (1) and (2) are well-founded orders. It is easily seen that the relation under (1) is a well-founded order. For (2) it is easily checked that (2) defines an order.
It remains to show that the ordered defined under (2) is well-founded. In the same way as in the proof of Theorem 2.1 an equivalence relation $\equiv$ can be defined. $\prec$ will not distinguish equivalent indexed literals under this relation. Now $\equiv$ has only a finite number of equivalence classes. Because of this the ordering defined under (2) is well-founded, and hence the composition of (1) and (2) is well-founded.
Now the resolution game proceeds in exactly the same way as in the proof of Theorem 2.1, and a $\sqsubset$-ordered refutation of C can be extracted from this game in the same manner.

6 Conclusions and Future Work

We have shown that there exists a large class of non-liftable orderings, with which resolution and factoring is complete. We have proven that the $<_v$-order is complete for the E'+-class, which was an open problem in ([FLTZ93]). We do not know to which extent the orderings are compatible with subsumption. Also we do not know what happens when condition SUBST in Theorem 2.1 is dropped. Counterexample 4.8 cannot be reproduced.

7 Acknowledgements

I would like to thank Trudie Stoute for her advice on English and Tanel Tammet for his improvements in the formulation of theorem 2.1.

References

[Baum92] P. Baumgartner, An ordered theory calculus, in LPAR92, Springer Verlag, Berlin, 1992.

[BG90] L. Bachmair, H. Ganzinger, On restrictions of ordered paramodulation with simplification, CADE 10, pp 427-441, Keiserslautern, Germany, Springer Verlag, 1990.

[Bezem90] M. Bezem, Completeness of resolution revisited, Theoretical computer science 74, pp. 227-237, 1990.

[Boyer71] R.S. Boyer, Locking: A restriction of resolution, Ph. D. Thesis, University of Texas at Austin, Texas 1971.

[ChangLee73] C-L. Chang, R. C-T. Lee, Symbolic logic and mechanical theorem proving, Academic Press, New York 1973.

[FLTZ93] C. Fermüller, A. Leitsch, T. Tammet, N. Zamov, Resolution methods for the decision problem, Springer Verlag, 1993.

[Joy76] W.H. Joyner, Resolution Strategies as Decision Procedures, J. ACM 23, 1 (July 1976), pp. 398-417.

[KH69] R. Kowalski, P.J. Hayes, Semantic trees in automated theorem proving, Machine Intelligence 4, B. Meltzer and D. Michie, Edingburgh University Press, Edingburgh, 1969.

[Nivelle94b] H. de Nivelle, Resolution games and non-liftable resolution orderings, Internal report 94-36, Department of Mathematics and Computer Science, Delft University of Technology.

[Robins65] J.A. Robinson, A machine oriented logic based on the resolution principle, Journal of the ACM, Vol. 12, pp 23-41, 1965.

[Tamm94] T. Tammet, Seperate orderings for ground and non-ground literals preserve completeness of resolution, unpublished, 1994.

[Zam72] N.K. Zamov: On a Bound for the Complexity of Terms in the Resolution Method, Trudy Mat. Inst. Steklov 128, pp. 5-13, 1972.