Generalized Completeness for SOS Resolution and its Application to a New Notion of Relevance

Fajar Haifani, Sophie Tourret, and Christoph Weidenbach

1 Max Planck Institute for Informatics, Saarland Informatics Campus, Saarbrücken, Germany
2 Graduate School of Computer Science, Saarbrücken, Germany
3 Université de Lorraine, CNRS, Inria, LORIA, Nancy, France

Abstract. We prove the SOS strategy for first-order resolution to be refutationally complete on a clause set $N$ and set-of-support $S$ if and only if there exists a clause in $S$ that occurs in a resolution refutation from $N \cup S$. This strictly generalizes and sharpens the original completeness result requiring $N$ to be satisfiable. The generalized SOS completeness result supports automated reasoning on a new notion of relevance aiming at capturing the support of a clause in the refutation of a clause set. A clause $C$ is relevant for refuting a clause set $N$ if $C$ occurs in every refutation of $N$. The clause $C$ is semi-relevant, if it occurs in some refutation, i.e., if there exists an SOS refutation with set-of-support $S = \{C\}$ from $N \setminus \{C\}$. A clause that does not occur in any refutation from $N$ is irrelevant, i.e., it is not semi-relevant. Our new notion of relevance separates clauses in a proof that are ultimately needed from clauses that may be replaced by different clauses. In this way it provides insights towards proof explanation in refutations beyond existing notions such as that of an unsatisfiable core.

1 Introduction

Shortly after the invention of first-order resolution [14] its first complete refinement was established: set-of-support (SOS) resolution [18]. The idea of the SOS strategy is to split a current clause set into two sets, namely $N$ and $S$ and restrict resolution inferences to have one parent from the set-of-support $S$. Wos et al. [18] proved the SOS strategy complete if $N$ is satisfiable. The motivation by Wos et. al. for the SOS strategy was getting rid of “irrelevant” inferences. If $N$ defines a theory and $S$ contains the negation of a conjecture (goal) to be refuted, the strategy puts emphasis on resolution inferences with the conjecture. This can be beneficial, because resolution is deductively complete (modulo subsumption) [11][13], i.e., resolution inferences solely performed on clauses from $N$ will enumerate all semantic consequences, not necessarily only consequences that turn out to be useful in refuting $N \cup S$. Even in more restrictive contexts, the SOS strategy can be shown complete, e.g., if $N$ is saturated by superposition and does not contain the empty clause, then the SOS strategy is also complete.
in the context of the strong superposition inference restrictions on $N$ and a set-of-support $S$.

In this paper, we generalize and sharpen the original completeness result for the SOS strategy: The resolution calculus with the SOS strategy is complete if and only if there is at least one clause in $S$ that is contained in a resolution refutation from $N \cup S$, Theorem[1]. The proof is performed via proof transformation. Any (non SOS) refutation from $N \cup S$ can be transformed into an SOS refutation with SOS $S$, if the original refutation contains at least one clause from $S$.

The generalized SOS completeness result supports our new notion of relevance that is meant to be a first stop towards explaining the gist of a refutation. A clause $C \in N$ is relevant if it is needed for any refutation of the clause set $N$. The clause $C$ is semi-relevant if there is a refutation from $N$ using $C$ and $C$ is irrelevant otherwise, Definition[12]. Applying our generalized SOS completeness result, a clause $C \in N$ is semi-relevant if and only if there is an SOS refutation from $N \setminus \{C\}$ with SOS $\{C\}$.

The interest in semi-relevant clauses comes from real-world applications. In an industrial scenario where different products are built out of a building set, the overall product portfolio is often defined by a set of clauses (rules). Roughly, every clause describes the integration of some part out of the building set in a product. Different proofs for the existence of some product correspond to different builds of the product. For example, answering a question like “Can we build car $x$ with part $y$?” from the automotive world boils down to the semi-relevance of the clauses defining part $y$ in a refutation showing the existence of a car $x$. All German car manufacturers maintain such clause sets defining their product portfolio[6,17].

Our new notion of relevance is related to other notions capturing aspects of a refutation. A minimal unsatisfiable core of an unsatisfiable clause set contains only semi-relevant clauses. The intersection of all minimal unsatisfiable cores is the set of relevant clauses. The notion of a minimal unsatisfiable core does not provide a test for semi-relevance of a specific clause. There are various notions from the description logic community related to unsatisfiable cores of a translation to first-order and/or to our notion of relevance[1][3][10]. An in-depth discussion of these relationships can be found in our description logic workshop paper[7]. The notion of relevant clauses is also related to what has been studied in the field of propositional satisfiability under the name of lean kernels[9][10]: Given an unsatisfiable set $N$ of propositional clauses, the lean kernel consists exactly of those clauses that are involved in at least one refutation proof of $N$ in the resolution calculus, and thus, in our terminology, the set of semi-relevant clauses. A different notion of relevance was previously defined in the context of propositional abduction[5]. The authors provide algorithms and complexity results for various abduction settings in the propositional logic context. In addition to the fact that our notion of relevance is defined with respect to first-order clauses, in their context of propositional abduction, if a propositional variable is relevant, it must be satisfiability preserving when added to the theory (clause set). In our case, if
a clause $C \in N$ is (semi-)relevant, then $N$ is unsatisfiable and $N \setminus \{C\}$ may be unsatisfiable as well.

The paper is organized as follows. After fixing some notations and notions at the beginning of Section 2, we introduce our proof transformation technique. First on an example, Figure 1, then in general. The following Section 3 proves important properties of the transformation, yielding our generalized completeness result for SOS, Theorem 11. We then link the SOS completeness result to our notion of semi-relevance in Section 4. The paper ends with a summary, a discussion of the contributions, and directions for future work, Section 5.

2 Resolution Proof Transformation

After fixing some common notions and notation, this section introduces our proof transformation technique. First on an example and afterwards on resolution refutations in general.

We assume a first-order language without equality where $N$ denotes a clause set; $C, D$ denote clauses; $L, K$ denote literals; $A, B$ denote atoms; $P, Q, R, T$ denote predicates; $t, s$ terms; $f, g, h$ functions; $a, b, c$ constants; and $x, y, z$ variables, all possibly indexed. Atoms, literals, clauses and clause sets are considered as usual. Clauses are disjunctions of literals. The complement of a literal is denoted by the function comp. Semantic entailment $|=\sigma$ considers variables in clauses to be universally quantified. Substitutions $\sigma, \tau$ are total mappings from variables to terms, where $\text{dom}(\sigma) := \{x | x \sigma \neq x\}$ is finite and $\text{codom}(\sigma) := \{t | x \sigma = t, x \in \text{dom}(\sigma)\}$. A renaming $\sigma$ is a bijective substitution. The application of substitutions is extended to literals, clauses, and sets/sequences of such objects in the usual way. The function $\text{mgu}$ denotes the most general unifier of two terms, atoms, literals if it exists. We assume that any $\text{mgu}$ of two terms or literals does not introduce any fresh variables and is idempotent.

The resolution calculus consists of two inference rules: Resolution and Factoring [14, 15]. The rules operate on a state $(N, S)$ where the initial state for a classical resolution refutation from a clause set $N$ is $(\emptyset, N)$ and for an SOS refutation with clause set $N$ and initial SOS $S$ the initial state is $(N, S)$. We describe the rules in the form of abstract rewrite rules operating on states $(N, S)$.

As usual we assume for the resolution rule that the involved clauses are variable disjoint. This can always be achieved by applying renamings to fresh variables.

**Resolution** $\quad (N, S \uplus \{C \land K\}) \Rightarrow_{\text{RES}} (N, S \uplus \{C \land K, (D \lor C)\sigma\})$

provided $(D \lor L) \in (N \cup S)$ and $\sigma = \text{mgu}(L, \text{comp}(K))$

**Factoring** $\quad (N, S \uplus \{C \land L \land K\}) \Rightarrow_{\text{RES}} (N, S \uplus \{C \land L \land K\} \cup \{(C \lor L)\sigma\})$

provided $\sigma = \text{mgu}(L, K)$

The clause $(D \lor C)\sigma$ is called the result of a Resolution inference between its parents. The clause $(C \lor L)\sigma$ is called the result of a Factoring inference of its parent. A sequence of rule applications $(N, S) \Rightarrow_{\text{RES}} (N, S')$ is called a
resolution derivation. It is called an SOS resolution derivation if $N \neq \emptyset$. In case $\bot \in S'$ it is called a (SOS) resolution refutation.

**Theorem 1 (Soundness and Refutational Completeness of (SOS) Resolution [14, 18]).** Resolution is sound and refutationally complete [14]. If for some clause set $N$ and initial SOS $S$, $N$ is satisfiable and $N \cup S$ is unsatisfiable, then there is a derivation of $\bot$ from $(N, S)$ [18].

Where a resolution derivation $(N, S) \Rightarrow^* \text{RES} (N, S')$ shows how new clauses can be derived from $(N, S)$, a deduction presents the minimal derivation of a single clause, e.g., the empty clause $\bot$ in case of a refutation. For deductions we require every clause to be used exactly once, so deductions always have a tree form. This is a purely technical restriction, see Corollary 5, that facilitates our deduction transformation technique that then needs not to take care of variable renamings except for input clauses.

**Definition 2 (Deduction).** A deduction $\pi_N = [C_1, \ldots, C_n]$ of a clause $C_n$ from some clause set $N$ is a finite sequence of clauses such that for each $C_i$ the following holds:

1.1 $C_i$ is a renamed, variable-fresh version of a clause in $N$, or
1.2 there is a clause $C_j \in \pi_N$, $j < i$ s.t. $C_i$ is the result of a Factoring inference from $C_j$, or
1.3 there are clauses $C_j, C_k \in \pi_N$, $j < k < i$ s.t. $C_i$ is the result of a Resolution inference from $C_j$ and $C_k$,

and for each $C_i \in \pi_N$, $i < n$:

2.1 there exists exactly one factor $C_j$ of $C_i$ with $j > i$, or
2.2 there exists exactly one $C_j$ and $C_k$ such that $C_k$ is a resolvent of $C_i$ and $C_j$ and $i, j < k$.

We omit the subscript $N$ in $\pi_N$ if the context is clear.

A deduction $\pi'$ of some clause $C \in \pi$, where $\pi, \pi'$ are deductions from $N$ is a subderivation of $\pi$ if $\pi' \subseteq \pi$, where for the latter subset relation we identify sequences with multisets. A deduction $\pi_N = [C_1, \ldots, C_{n-1}, \bot]$ is called a refutation.

Note that variable renamings are only applied to clauses from $N$ that are introduced in the deduction are variable disjoint.

**Definition 3 (SOS Deduction).** A deduction $\pi_{N \cup S} = [C_1, \ldots, C_n]$ is called an SOS deduction if the derivation $(N, S_0) \Rightarrow^* \text{RES} (N, S_m)$ is an SOS derivation where $C'_1, \ldots, C'_m$ is the subsequence from $[C_1, \ldots, C_n]$ with input clauses removed, $S_0 = S$, and $S_{i+1} = S_i \cup C'_{i+1}$.

**Definition 4 (Overall Substitution of a Deduction).** Given a deduction $\pi$ of a clause $C_n$ the overall substitution $\tau_{\pi,i}$ of $C_i \in \pi$ is recursively defined by

1. if $C_i$ is a factor of $C_j$ with $j < i$ and $\text{mgu} \sigma$, then $\tau_{\pi,i} = \tau_{\pi,j} \circ \sigma$, 

2 if $C_i$ is a resolvent of $C_j$ and $C_k$ with $j < k < i$ and mgu $\sigma$, then $\tau_{\pi,i} = (\tau_{\pi,j} \circ \tau_{\pi,k}) \circ \sigma$, 
3 if $C_i$ is an initial clause, then $\tau_{\pi,i} = \emptyset$, 
and the overall substitution of the deduction is $\tau_{\pi} = \tau_{\pi,n}$. We omit the subscript $\pi$ if the context is clear.

Overall substitutions are well-defined, because clauses introduced from $N$ into the deduction are variable disjoint and each clause is used exactly once in the deduction. A grounding of an overall substitution $\tau$ of some deduction $\pi$ is a substitution $\tau_\delta$ such that codom($\tau_\delta$) only contains ground terms and dom($\delta$) is exactly the variables from codom($\tau$).

**Corollary 5 (Deduction Refutations versus Resolution Refutations).** There exists a resolution refutation $(N,S) \Rightarrow_{\text{RES}}^* (N',S' \cup \{\bot\})$ if and only if there exists a deduction refutation $\pi(N,S) = [C_1, \ldots, C_{n-1}, \bot]$ where $C_i \in (N \cup S')$ for all $i$, modulo variable renaming.

We prove the generalized completeness result of SOS by transforming non-SOS refutations into SOS refutations. For illustration of our proof transformation technique, consider the below unsatisfiable set of clauses $N$. Literals are labeled in $N$ by a singleton set of a unique natural number \{12\}. We will refer to the literal labels during proof transformation in order to identify resolution and factorization steps. The labels are inherited in a resolution inference and united for the factorized literal in a factoring inference. See the factoring inference on clause (3), Figure 1.

$$N = \{(1):\{1\} \neg Q(x_3,f(a)) \lor \{2\} P(f(a)), \ (2):\{3\} \neg P(x_4) \lor \{4\} \neg Q(b,x_4), \ (5):\{5\} \neg Q(b,a) \lor \{6\} Q(x_1,f(x_6)), \ (6):\{7\} Q(b,x_2) \lor \{8\} R(x_2) \lor \{9\} T(c,x_1), \ (9):\{10\} \neg R(x_5), \ (11):\{11\} \neg T(c,b)\}$$

Figure 1 shows a resolution refutation $\pi = [(5),(6),(7),(1),(2),(3),(4),(8),(9),(10),(11),(12)]$ from $N$. This resolution refutation is also an SOS refutation with SOS $S = \{(2),(5)\}$ and remaining clause set $N \setminus S$. It is not an SOS refutation with SOS $S = \{(5)\}$ and the remaining clause set $N \setminus S$ because the resolution step between clauses (1) and (2) is not an SOS step. The shaded part of the tree belongs to an SOS deduction with $S = \{(5)\}$.

The transformation identifies a clause closest to the leaves of the tree, obtained by resolution, that has one parent that can be derived by the SOS strategy, but the other parent is not in the SOS nor an input clause. For our example with starting SOS $S = \{(5)\}$ this is clause (8). The parent (7) can be derived via SOS from $S$ but the other parent (4) is not part of an SOS derivation. The overall grounding substitution of $\pi$ is $\tau = \{x_1 \mapsto b, x_2 \mapsto a, x_3 \mapsto b, x_4 \mapsto f(a), x_5 \mapsto a, x_6 \mapsto a\}$. Now the idea of a single transformation step is to perform the
The deduction can then be continued the same way it was done in \( \pi \). From input clauses need to be resolved: here the literals \( \{3\} \). Now \( \pi' \) is simulated with the resolved input clauses, see Figure 3. Finally, the rest of the deduction yielding clause \( 8 \) is simulated with the resolved input clauses, see Figure 3. To this end the resolution step on the labelled literal \( \{1, 4\} \) and the respective literal \( \{6\} \) of the SOS derivable clause \( 7 \) already on the respective literals from the input clauses yielding \( 8 \), here clauses \( 1 \) and \( 2 \). To this end the derivation \([5], 6, 7\) is copied with fresh variables, see Figure 2, yielding the clauses \( 7 \) and \( 7' \) used in the refutation \( \pi' \) below, see also Figure 3.

\[
\begin{align*}
(5): [5] & \Rightarrow Q(b, a) \lor (6) \Rightarrow Q(x_1, f(x_0)) \\
(6): [7] & \Rightarrow Q(b, x_2) \lor (8) \Rightarrow R(x_2) \lor (9) \Rightarrow T(c, x_1) \\
(7): [6] & \Rightarrow Q(x_1, f(x_0)) \lor (8) \Rightarrow R(a) \lor (9) \Rightarrow T(c, x_1) \\
(8): [4] & \Rightarrow Q(b, f(a)) \\
(9): [10] & \Rightarrow R(x_5) \\
(10): [9] & \Rightarrow T(c, b) \\
(11): [11] & \Rightarrow T(c, b) \\
(12): \bot
\end{align*}
\]

**Fig. 1.** Refutation of \( \pi \) of \( N \)

The two freshly renamed copies \( 7 \) and \( 7' \) are resolved with the respective input clauses \( 1 \) and \( 2 \). Finally, the rest of the deduction yielding clause \( 8 \) is simulated with the resolved input clauses, see Figure 3. Now \( 8'' \) is exactly clause \( 8 \) from the original deduction \( \pi \), but \( 8'' \) is derived by an SOS deduction. The deduction can then be continued the same way it was done in \( \pi \) and in this case will already yield an SOS refutation.

\[
\pi' = [5], (6), (7), (5'), (6'), (7'), (1), (1'), (2), (2'), (8'), (8''), (8'''), (9), (10), (11), (12)].
\]

The example motivates our use of literal labels. Firstly, they tell us which literals from input clauses need to be resolved: here the literals \( \{1\} \Rightarrow Q(x_3, f(a)) \) and \( \{4\} \Rightarrow Q(b, x_4) \) that are factorized in \( \pi \) to \( \{1, 4\} \Rightarrow Q(b, f(a)) \). Secondly, they guide additional factoring steps in \( \pi' \) during the simulation of the non-SOS part from \( \pi \): here the factoring between the two literals labelled \( 8 \) in clause \( 8' \) and...
the two literals with label \{9\} in clause (8\textquoteright\textquoteright). The transformation always works because the overall grounding substitution of the initial refutation \(\pi\) is preserved by the transformation. It just needs to be extended to the extra variables added by freshly renamed copies of clauses.

The above example shows the importance of keeping track of the occurrences of literals in a deduction. A labeled literal is a pair \(ML\) where \(M\) is a finite non-empty set of natural numbers called the label and \(L\) is a literal. We identify literals with labeled literals and refer explicitly to the label of a labeled literal by the function \(lb\). The function \(lb\) is extended to clauses via union of the respective literal labels. We extend the notion of a clause to that of a labeled clause.

In general, we need to identify the parts of a deduction that are already contained in an SOS deduction, this is called the partial SOS of a deduction, Definition\[6]\]. Then this information can be used to perform the above transformation on any deduction \(\pi\).

**Definition 6 (PSOS of a Deduction).** Let \(\pi\) be a deduction from \(N \cup S\), then the partial SOS (PSOS) \(O^*\) of \(\langle \pi, N, S\rangle\) is defined as \(O^* = \bigcup_{i=0}^m O^i\), where \(O^0 = S\), \(O^{i+1} = O^i \cup \{C_j\}\) provided \(C_j \in \pi\), \(C_j \notin O^i\) and \(C_j\) is either the factor of some clause in \(O^i\) or the resolvent of two clauses in \(\pi\) where at least one parent is from \(O^i\), and where \(O^{m}\) is such that there is no longer such a \(C_j\) in \(\pi\).

The partial SOS is well-defined because the resulting \(O^*\) is independent of the sequence \(O^i\) used. For example, for the deduction \(\pi\) from \(N\) presented in

![Fig. 3. The new SOS deduction yielding a copy of clause (8) on any deduction \(\pi\).](image)

...
Figure 1 the set $O^* = \{5\} \cup (6) \cup (7)$ is the PSOS of $\langle \pi, N, \{5\} \rangle$. Next we present a criterion when the PSOS of a deduction actually signals an SOS deduction.

**Lemma 7 (SOS Deduction).** Let $O^*$ be the PSOS of $\langle \pi, N, S \rangle$. Then $\pi$ is an SOS deduction if $O^* \setminus S = \pi \setminus (N \cup S)$, i.e., all inferred clauses in $\pi$ are contained in $O^*$.

**Proof.** Let $\pi_{NLS} = [C_1, \ldots, C_n]$ and $[C'_1, \ldots, C'_m]$ be the subsequence of $\pi_{NLS}$ with input clauses removed. Let $O^*$ be the PSOS of $\langle \pi, N, S \rangle$. Then $[C'_1, \ldots, C'_m] = O^* \setminus S = \pi \setminus (N \cup S)$ by assumption. We show that $(N, S^0) \Rightarrow^* \text{RES} (N, S^m)$ is an SOS derivation, following Definition 3 by induction on $m$. If $m = 0$ then $\pi$ only consists of input clauses and there is nothing to show. For the case $m = 1$, the clause $C'_1$ is the result of a factoring inference from $S$ or the result of a resolution inference from $N \cup S$ such that at least one parent is in $S$ as for otherwise $C'_1 \not\in (O^* \setminus S)$. So $(N, S^0) \Rightarrow^* \text{RES} (N, S^0 \cup \{ C'_1 \})$ is an SOS derivation.

For the induction case, assume the property holds for $i$. If $C'_{i+1}$ is the result of a factoring inference, then its parent $C''$ is contained in $S^i$ because otherwise $C'' \in N$ because $\pi$ being a deduction, and, therefore $C'_{i+1} \not\in (O^* \setminus S)$, a contradiction. If $C'_{i+1}$ is the result of a resolution inference, then again all its parents are contained in $N \cup S^i$ because $\pi$ is a deduction. If both parents are from $N$, then $C'_{i+1} \not\in (O^* \setminus S)$, a contradiction. So, by the induction hypothesis, $(N, S^0) \Rightarrow^* \text{RES} (N, S^0) \Rightarrow^* \text{RES} (N, S^{i+1})$ is an SOS derivation.

The rest of this section is devoted to describing the transformation in detail. In the next section, we then prove the new completeness result for SOS.

Let $\pi$ be a label-disjoint deduction from $N \cup S$ and let $C_k \in \pi$ be a clause of minimal index such that $C_k$ is the result of a resolution inference from clauses $C_j \in O^*$ and $C_i \not\in (N \cup O^*)$. Let $\tau$ be an overall ground substitution for $\pi$. We transform $\pi$ into $\pi'$ by changing the deduction of $C_i$ such that the overall deduction gets “closer” to an SOS derivation and preserves $\tau$. Let

\[
C_j = C'_j \lor L \\
C_i = C'_i \lor K \\
C_k = (C''_i \lor C'_j)\sigma
\]

where $\sigma = \text{mgu}(K, \text{comp}(L))$. Without loss of generality we assume that

\[
\pi = [C_1, \ldots, C_i, C_{i+1}, \ldots, C_j, C_k, C_{k+1}, \ldots, C_n]
\]

where $[C_1, \ldots, C_i]$ and $[C_{i+1}, \ldots, C_j]$ are subdeductions of $\pi$, and the prefixes of these sequences are exactly the introduced renamed copies of input clauses from $N$ that are used to derive $C_i$ and $C_j$, respectively. The transformed derivation will be

\[
\pi' = [C'_{i+1}, \ldots, C'_j, \ldots, C''_i, \ldots, C'_j, D_1, \ldots, D_l, C'_k, \ldots, C_n]
\]

where

---

4 Here we refer to the removal of all input clauses from $O^*$ and $\pi$, respectively.
(a) the subsequences \([C_{i+1}^0, \ldots, C_j^0]\) are freshly variable-renamed copies of the sequence \([C_{i+1}, \ldots, C_j]\) where \(m = |\text{lb}(K)|\). For the copies \([C_{i+1}^0, \ldots, C_j^0]\) we keep the labels of literals of the original sequence \([C_{i+1}, \ldots, C_j]\) for reference in the transformation. The clauses \(C_j^0\) are decomposed into \(C_j^0 \lor L'\), in the same way that the clause \(C_j\) is decomposed into \(C_j \lor L\). Thus, for each clause from \(N\) in the sequence \([C_{i+1}, \ldots, C_j]\) containing a literal \(K'\) with \(\text{lb}(K') \subseteq \text{lb}(K)\) we add a deduction deriving a renamed copy of \(C_j^0\): let \(\delta^p\) be the renaming substitution from the old to the freshly renamed sequence, then we extend \(\tau\) to \(\tau'\) as follows: \(\tau'_0 = \tau, \tau'_{p+1} = \tau'_p \circ \{u \mapsto v| u \in \text{dom}(\delta^{p+1}), v = x\}\) for \(1 \leq p \leq m\) yielding the overall new grounding substitution \(\tau' = \tau'_m\) for \(\tau'\).

(b) the clauses \(D_1, \ldots, D_t\) are generated by simulating the deduction \([C_1, \ldots, C_t]\) eventually producing \(C_k\) up to possible variable renamings: Let \(C_p\) be the current clause out of this deduction and let \(D_1, \ldots, D_q\) be the clauses generated so far until \(C_p\): 

(i) if \(C_p\) is an input clause not containing a literal \(K'\) with \(\text{lb}(K') \subseteq \text{lb}(K)\), then \(D_{q+1} = C_p\) and we associate \(D_{q+1}\) with \(C_p\); 

(ii) if \(C_p\) is an input clause containing a literal \(K'\) with \(\text{lb}(K') \subseteq \text{lb}(K)\), then \(D_{q+1} = C_p\) and \(D_{q+2}\) is the resolvent between \(D_{q+1}\) and a so far unused clause \(C_j^0\) on the literals \(K' \in D_{q+1}\) and \(L' \in C_j^0\) where \(\text{lb}(K') \subseteq \text{lb}(K)\) and \(\text{lb}(L') = \text{lb}(L)\) and we associate \(D_{q+2}\) with \(C_j^0\); 

(iii) if \(C_p\) is the resolvent between two clauses \(C_{i'}, C_{j'}\) then we perform the respective resolution step between the associated clauses and respective associated literals from \(D_{q'}, D_{q'}\) yielding \(D_{q+1}\) and associate \(D_{q+1}\) with \(C_p\); 

(iv) if \(C_p\) is the factor on some literal \(K'\) with \(\text{lb}(K') \subseteq \text{lb}(K)\), then we perform the respective factoring steps \(D_{q+1}, \ldots, D_{q+s}\) for respective literals with labels from \(C_{j'}\), where \(s = |C_{j'}|\) and we associate \(D_{q+s}\) with \(C_p\); 

(v) if \(C_p\) is the factor on some literal \(K'\) with \(\text{lb}(K') \not\subseteq \text{lb}(K)\), then we perform the respective factoring step on the respective literals with identical labels from clause \(D_{q'}\) yielding \(D_{q+1}\) and we associate \(D_{q+1}\) with \(C_p\); 

(c) the clauses \(C_{k+1}^0, \ldots, C_n^0\) are obtained by simulating the generation of clauses \(C_{k+1}, \ldots, C_n\) where \(C_k\) is substituted with \(D_1\).

Note that by assumption, the generation of clauses \(C_{k+1}, \ldots, C_n\) does not depend on clauses \(C_1, \ldots, C_{k}, C_{k+1}, \ldots, C_j\) but only on \(C_k\) and the input clauses. We will prove that \(C_k \tau = C_k \tau' = D_1 \tau'\) which is then sufficient to prove \(C_n \tau = C_n \tau' = C_n' \tau'\) and for the above to be well-defined. In general, the clause \(D_1\) is not identical to \(C_k\) because we introduce fresh variables in \(\pi'\) and do not make any specific assumptions on the unifiers used to derive \(D_1\).

Mapping the transformation to our running example, Figure 1 \(C_j = (7), C_4 = (4),\) and \(C_k = (8)\). We need two copies of \((7)\) because \(K = \{1, 4\}\) and \(L = \{6\} Q\{x_1, f(x_0)\}\) so \(m = |\{1, 4\}| = 2\) and \(L = \{6\} Q\{x_1, f(x_0)\}\).
3 A Generalized Completeness Proof for SOS

In this section, we prove that repeated applications of the transformation introduced in the previous section can actually transform an arbitrary deduction into an SOS deduction, given that at least one clause from the SOS occurs in the original deduction. Firstly, we show that associated clauses of the transformed deduction preserve main properties of the original deduction. The extended substitution is identical to the original substitution on old clauses and the changed part of the deduction ends in exactly the same clause.

Lemma 8 (Properties of Associated Clauses). Let $C_j, C_i, L, K, \pi, \pi', \tau, \tau'$ be as defined in [1], [2], and [3], page 8. For each clause $C$ out of $[C_1, \ldots, C_i]$ and clause $D$ associated with $C$:

1. $C\tau = C\tau'$,
2. $K\tau' = L\tau'$ if $lb(K') \subseteq lb(L')$ for any $K', L'$ occurring in either $\pi$ or $\pi'$,  
3. $lb(C) \setminus lb(K) = lb(D) \setminus lb(C')$ and $lb(C') \subseteq lb(D)$ if there is $K' \in C$ with $lb(K') \subseteq lb(K)$,
4. $C\tau \setminus \{K'\tau \in C \mid lb(K') \subseteq lb(K)\} = D\tau' \setminus \{L'\tau' \in D\tau' \mid lb(L') \subseteq lb(C')\}$ and $C'\tau' \subseteq D\tau'$ if there is $K' \in C$ with $lb(K') \subseteq lb(K)$,
5. $C_k\tau = D_1\tau'$.

Proof. [1] By definition of $\tau'$ the additional variables in $\tau'$ do not occur in $C$ while $\tau$ is identical to $\tau$ on the variables of $C$, hence $C\tau = C\tau'$.
[2] By induction on the generation of $\tau'$. For the base case, every literal occurring in $N \cup S$ has a unique label and any renamed clause $C'_{m}$ for some $C_{m} \in (N \cup S)$ has the labels kept. So, for any two literals $K'$ and $L'$ in any non invariant clauses in $\pi$ and $\pi'$, $K'\tau' = L'\tau'$ when the labels are equal. For the induction step, for inferred clauses, $lb(K') = lb(L')$ happens when the label of $K'$ is inherited from $L'$ through an inference. The inference uses an mgu which is compatible with $\tau'$ due to $\tau'$ being an overall ground substitution, hence $K'\tau' = L'\tau'$.
[3] We prove this property by induction on the length of the derivation $[C_1, \ldots, C_i]$. Let $C = C_p, 1 \leq p \leq i$, and let $D_1, \ldots, D_q$ be the clauses generated until $C_{p-1}$ for which, by the induction hypothesis the property already holds.

(i) If $C$ is an input clause not containing a literal $K'$ with $lb(K') \subseteq lb(K)$, we have $C = C_p = D_{q+1} = D$ and $\{K' \in C \mid lb(K') \subseteq lb(K)\} = \{L' \in D_q\tau' \mid lb(L') \subseteq lb(C')\} = \emptyset$.

(ii) If $C$ is an input clause containing a literal $K'$ with $lb(K') \subseteq lb(K)$ then $D = D_{q+2}$ results from a resolution inference between $C = C_p$ and an unused $C_{j'}$ on the literals $K'$ and $L' \in C_{j'}$ with $lb(L') = lb(L)$. Let $C = C' \setminus K'$. Then $D\tau' = (C' \setminus C'_{j'})\tau'$ and hence $lb(C) \setminus lb(K) = lb(D) \setminus lb(C'_{j'})$ because $lb(C) \cap lb(C'_{j'}) = \emptyset$ as $\pi$ is a label-disjoint deduction and $lb(C_{j'}) = lb(C'_{j'})$ by construction.

(iii) If $C$ is a resolvent of $C_{i'} = C'_{i'} \setminus L''_i$ and $C_{j'} = C'_{j'} \setminus L''_j$, on literals $L''_i, L''_j$, then $C\tau = C_{i'}\tau \setminus C_{i'}\tau$, and $D_{q+1}$ is a resolvent of some $D_{q'} = D'_{q'} \setminus L''_q$ and $D_{q''} = D'_{q''} \setminus L''_q$ associated with $C_{i'}$ and $C_{j'}$ respectively. We have
If deduction π
This means
2. Towards contradiction, suppose π
lb(L′) = lb(L′′)
and none of these literals has a
label from lb(K) or lb(C′′).
Hence, the conjecture holds by the induction
hypothesis.
(iv) If C results from a factoring on K′ from C′−1, we get D′+s by a sequence
of s factoring inferences from D′+1 associated with C′−1. Any factorings
on C′−1 and D′+1 do not change literal labels because we factorize literals
of identical label. So, this property holds by the induction hypothesis. This
holds regardless of whether lb(K′) ⊆ lb(K).

From Lemma 8.1 we know that lb(C) \ lb(K) = lb(D) \ lb(C′′) and lb(C′′) ⊆
ib(D) if there is K ∈ C with lb(K) ⊆ lb(K′). Since the labels coincide, using
Lemma 8.2 we have C′ \ {K′ ∈ C′ | lb(K′) ⊆ lb(K)} = D′ \ {L′ ∈ D′ | lb(L′) \ lb(C′′)} and C′′ ⊆ D′ if there is K′ ∈ C with lb(K′) ⊆ lb(K). This
hypothesis holds by applying Lemma 8.1 on literals and clauses from π in the
equation.

The clause Ck is the result of a resolution inference between Cj and Ck upon
K and L: Ckτ = Cjτ ∪ Cjτ. By translation and because \{K′ ∈ Cj | lb(K′) ⊆
ib(K)} = \{K\}, the clause Cj is associated with D′ ∈ K and Cj ∩ \{Kτ\} =
D′ \ \{L′ ∈ D′ | lb(L′) \ lb(C′′)\}. Since C′′ ⊆ Cjτ = Cjτ \ \{Lτ\}, we have
\{L′ ∈ D′ | lb(L′) \ lb(L)\} for some L′ ∈ Cjτ. Thus, Cjτ \ \{Lτ\} = Cj \ \{Lτ\}.
Furthermore, Ckτ = Cjτ \ \{Kτ, Lτ\} = Ck \ \{Kτ, Lτ\}.
So Ckτ \ \{Kτ, Lτ\} = D′ \ \{D′τ \ Cj \ \{Lτ\}\} = D′ τ \ Cj \ \{Lτ\}. We can add
Cjτ \ \{Lτ\} to both sides and get Cjτ \ Cjτ \ \{Kτ, Lτ\} ⊆ D′τ. In
addition, since lb(K) ⊆ lb(K), this means Cjτ = C′′ τ ⊆ D′τ. Therefore
Ckτ = Ckτ ∪ Cjτ \ \{Kτ, Lτ\} = D′τ.

Next we need a well-founded measure that decreases with every transformation
step and in case of reaching its minimum signals an SOS deduction. Given a
clause set N and an initial SOS S, the SOS measure of a deduction π is µ(π) where
µ(π) = ∑ Cj ∈ π µ(Cj, π) and µ(Cj, π) = 0 if Cj ∈ N \ O∗ otherwise µ(Cj, π) = 1.

Lemma 9 (Properties of µ). Given a clause set N, an initial SOS S, and a
deduction π that contains at least one resolution step,
1. µ(π) ≥ 0, and
2. if µ(π) = 0 then π is an SOS deduction.

Proof. 1. Obvious.
2. Towards contradiction, suppose π = [C1, ..., Cn] is not an SOS deduction.
This means O∗ \ S ⊆ π \ (N ∪ S) by Lemma 7. Consider a clause Cj ∈ (π \ (N ∪ S)) \ (O∗ \ S) of minimal index. Then Cj must be the result of an inference on
some Cj and Ck such that both are not in O∗. This means Cj ∈ (N ∪ O∗). For
this clause, µ assigns a nonzero value: µ(Cj, π) > 0. Therefore, µ(π) ≠ 0. 3.
Lemma 10 (Properties of the Transformation). Given a deduction $\pi$ of a clause $C_n$ from $N \cup S$ that contains at least one resolution step such that $\pi \cap S \neq \emptyset$, an overall ground substitution $\tau$ of $\pi$ and the transformed deduction $\pi'$ of a clause $C'_n$ as defined in \[1\], \[2\], and \[3\] with overall ground substitution $\tau'$, we have:

1. $\pi'$ is a deduction from $N \cup S$,
2. $C_n \tau = C'_n \tau'$, and
3. $\mu(\pi') < \mu(\pi)$.

Proof. We show that $\pi'$ is a deduction following Definition 2. These properties will be carried over from $\pi$. Observe that, if $\pi_1$ is a deduction of $C_k$ from $N \cup S$ and $\pi_2$ is a deduction from $N \cup S \cup \{C_k\}$ using $C_k$ only once, their concatenation $\pi_1 \circ \pi_2$ is a deduction from $N \cup S$. Firstly, the subsequences $[C'_{i+1}, \ldots, C'_{j}]$ are deductions of $C'_j$ from $N \cup S$ since they are only the renamed copies of the subderivation $[C_{i+1}, \ldots, C_j]$ of $\pi$. Secondly, the subsequence $[C_k, \ldots, C_n]$ is a deduction of $C_n$ from $N \cup S \cup \{C_k\}$ since the clauses after $C_k$ do not use any clauses before $C_k$ by the way $\pi$ is represented as a sequence. Now, by showing that $[C'_{i+1}, \ldots, C'_{j}, D_1, \ldots, D_l, C_d]$ is a deduction of $C_k$ from $N \cup S \cup \{C'_{j}, \forall i \in [1, m]\}$, the sequence $[D_1, \ldots, D_l]$ would then connect the initial copied sequences and the tailing subsequence. Each $C'_j$ is used for exactly one resolution inference producing some $D_q$, the other required clauses are copied, and the later resolution and factoring steps in $[D_1, \ldots, D_l]$ are sound while the deduction properties of $[C_1, \ldots, C_l]$ are preserved in its associated clauses: for an inference where $C_{p'}$ (and $C_{p''}$) generates $C_p$, we have a unique inference between their associated clauses $D_{q'}, (D_{q''}) D_{q+1}$ where $D_{q'}$ (and $D_{q''}$) generates $D_{q+1}$, possibly with additional factoring inferences in between. If $C_p$ is an input clause not containing a literal $K'$ with $\text{lb}(K') \subseteq \text{lb}(K)$, then $D_{q+1} = C_p \in N$. The clause $D_{q+1}$ is used in $\pi'$ as $C_p$ is used in $\pi$; if $C_p$ is an input clause containing a literal $K'$ with $\text{lb}(K') \subseteq \text{lb}(K)$, the resolution between $D_{q+1}$ and a so far unused clause $C'_j$ is sound as $K'$ and $\text{comp}(L')$ are unifiable by $\tau'$. Here, all $C'_j$ will be eventually used as there are $m = |\text{lb}(K)|$ literals in the clauses from $N$; if $C_p$ is the resolvent between two clauses $C_{p'}, C_{p''}$ then the respective resolution step between the associated clauses $D_{q'}, D_{q''}$ upon the respective associated literals $K'$ and $L'$ is sound because we can get $K' \tau' = \text{comp}(L') \tau'$ using Lemma 8 if $C_p$ is the factor on some literal $K'$ with $\text{lb}(K') \subseteq \text{lb}(K)$, then the respective factoring steps $D_{q+1}, \ldots, D_{q+s}$ are also sound: each pair of the $s$ associated literals $M$ and $M'$ from $C_j'$ and $C_j''$ are unifiable because $M \tau' = M' \tau'$; if $C_p$ is the factor of $C_{p-1}$ upon some literal $K'$ and $L'$ with $\{\text{lb}(K'), \text{lb}(L')\} \subseteq \text{lb}(K)$, the respective factoring step on the associated clause $D_{q'}$ is also sound by Lemma 8. Therefore $\pi$ is a deduction from $N \cup S$.

By Lemma 8, $C_k \tau = D_l \tau'$. The derivation of clauses $C_k, C_{k+1}, \ldots, C_n$ only depends on the input clauses by assumption. By an inductive argument we get $C_{k+1} \tau = C'_{k+1} \tau'$ yielding $C_n \tau = C'_n \tau'$.

The clauses in $[C'_{i+1}, \ldots, C'_{j}]$ have the measure 0 as their original ones in $[C_{i+1}, \ldots, C_j]$ because they are in $N \cup O^*$. The clauses in $[C_k, \ldots, C_n]$ also retain their original measures. The clauses in $[D_1, \ldots, D_l]$ are s.t. $\sum_{k=1}^l \mu(\pi', D_k) < \sum_{k=1}^l \mu(\pi, D_k)$.
\[ \Sigma_i \mu(\pi', C_k). \] More specifically, any \( C \in [C_1, \ldots, C_i] \) that is not in \( N \cup O^* \) (with measure \( \mu(C, \pi) \geq 1 \)) and containing \( K' \) with \( \text{lb}(K') \subseteq \text{lb}(K) \) is associated with \( D_q \in O^* \setminus N \) having the measure \( \mu(D_q, \pi') = 0 \), while all other clauses in \( [D_1, \ldots, D_i] \) are either copied from \( \pi \) with the same measure as before or new in \( \pi' \) but have the measure 0.

By induction on the length of the sequence \([C_1, \ldots, C_i]\) we prove the following property: if \( D \) is associated with a clause \( C \in [C_1, \ldots, C_i] \) and \( C \) contains some literal from \( \{K' | \text{lb}(K') \subseteq \text{lb}(K)\} \), then \( D \in N \cup O^* \) and \( \mu(D, \pi') = 0 \). Let \( C = C_p \). Let \( D_1, \ldots, D_q \) be the clauses generated until \( C_{p-1} \) s.t. the property already holds.

(i) If \( C_p \) is an input clause with no literals in \( \{K' | \text{lb}(K') \subseteq \text{lb}(K)\} \), it is associated with \( D_q = C_p \) s.t. \( \mu(C_p, \pi) = \mu(D_q, \pi') = 0 \);

(ii) If \( C_p \) is an input clause containing \( \{K' | \text{lb}(K') \subseteq \text{lb}(K)\} \), it is resolved with some \( C_j' \in O^* \) resulting in \( D_{q+1} \in O^* \). Here we have \( \mu(C_p, \pi) = \mu(D_q, \pi') = 0 \);

(iii) If \( C_p \) is the resolvent between two clauses \( C_{i'}, C_{i''} \), then we perform the respective resolution step between the associated clauses \( D_{q'}, D_{q''} \) yielding the clause \( D_q \) associated with \( C_p \). If either \( C_{i'} \) or \( C_{i''} \) contains some literal from \( \{K' | \text{lb}(K') \subseteq \text{lb}(K)\} \) then \( C_p \) contains this literal as well and either \( D_{q'} \in O^* \) or \( D_{q''} \in O^* \) by the induction hypothesis. So, we get \( D_q \in O^* \) and \( \mu(D_q, \pi') = 0 \). Otherwise, \( \mu(D_q, \pi') = \mu(C_p, \pi) = 1 \);

(iv) If \( C_p \) is the factor of \( C_{p-1} \) on some literal \( K' \) with \( \text{lb}(K') \subseteq \text{lb}(K) \), then we have the respective factoring steps \( D_{q+1}, \ldots, D_{q+s} \) where \( D_{q+1} \) is associated with \( C_{p-1} \). By the induction hypothesis, \( D_{q+1} \in O^* \). Therefore \( D_{q+1}, \ldots, D_{q+s} \in O^* \) with \( \mu(D_{q+t}, \pi') = 0 \) for \( 1 \leq t \leq s \);

(v) If \( C_p \) is the factor of \( C_{p-1} \) (associated with \( D_q \)) on some literal \( K' \) and \( L' \) with \( \{\text{lb}(K'), \text{lb}(L')\} \not\subseteq \text{lb}(K) \), the factoring happens to the associated clauses in \( \pi' \) with similar measure.

Finally, by the choice of \( C_1, C_2, \) and \( C_k \), there must exist at least one \( C_p \) with some literal from \( \{K' | \text{lb}(K') \subseteq \text{lb}(K)\} \) but associated with some \( D \) such that \( D \in O^* \) from case (iii) or (iv) before. This also means \( \mu(D, \pi') = 0 \). The clause \( C_1 \) has this property as it contains \( K \). In addition, any \( C_p \) has a nonzero measure because \( C_i \not\in N \cup O^* \) and \( C_p \) is used to prove \( C_i \). Therefore, we have \( \mu(C_p, \pi) > \mu(D, \pi') = 0 \). As these clauses are never copied to \( \pi' \), \( \mu(\pi') < \mu(\pi) \).

Eventually, by an inductive argument we prove our main result.

**Theorem 11 (Generalized SOS Completeness).** There is an SOS resolution refutation from \((N, S)\) if and only if there is resolution refutation from \(N \cup S\) that contains at least one clause from \( S\).

**Proof.** “⇒”: Obvious: If there is no refutation from \( N \cup S \) using a clause \( S \) then there can also not be any SOS resolution refutation from \((N, S)\).

“⇐”: If there is a deduction refutation \( \pi \) from \( N \cup S \) that contains at least one clause from \( S \), then by an inductive argument on \( \mu \) it can be transformed into
an SOS deduction refutation with SOS $S$, and the result follows by Corollary $5$. If $\mu(\pi) = 0$ then $\pi$ is already an SOS deduction, Lemma $9$. For otherwise, we transform the deduction $\pi$ into a deduction $\pi'$ according to (1), (2), and (3). A refutation always contains at least one resolution step, so by Lemma $10$, $\pi'$ is also a refutation from $N \cup S$ and $\mu(\pi') < \mu(\pi)$. Eventually, $\pi'$ can be transformed into a label-disjoint deduction by assigning fresh labels to all used clauses from $N \cup S$.

As an example for the “$\Rightarrow$” direction consider the propositional logic clause set $N = \{P, \neg P\}$ and SOS $S = \{Q\}$. Obviously, there is no refutation of $N \cup S$ using $Q$ and there is no SOS refutation. Theorem $11$ also guarantees that the consecutive application of the proof transformation steps (1), (2), and (3), page $8$ results in an effective recursive procedure that transforms non-SOS refutations into SOS refutations.

4 A new Notion of Relevance

The idea of our notion of relevance is to separate clauses that are ultimately needed in a refutation proof called relevant, from clauses that are useful called semi-relevant, from clauses that are not needed called irrelevant.

Definition 12 (Relevance). Given an unsatisfiable set of clauses $N$, a clause $C \in N$ is relevant if for all deduction refutations $\pi$ of $N$ it holds that $C \in \pi$. A clause $C \in N$ is semi-relevant if there exists a deduction refutation $\pi$ of $N$ in which $C \in \pi$. A clause $C \in N$ is irrelevant if there is no deduction refutation $\pi$ of $N$ in which $C \in \pi$.

With respect to our example clause set $N$ from Section $2$ and its refutation, Figure $1$, clause (5) is semi-relevant but not relevant, because the clauses (1), (2), (6), (9), (11) are already unsatisfiable. The clauses (1), (2), (6), (9), (11) are all relevant.

Lemma 13 (Relevance). Given an unsatisfiable set of clauses $N$, the clause $C \in N$ is relevant if and only if $N \setminus \{C\}$ is satisfiable.

Proof. Obvious: if $N \setminus \{C\}$ is satisfiable there is no resolution refutation and since $N$ is unsatisfiable $C$ must occur in all refutations. If $C$ occurs in all refutations there is no refutation without $C$ so $N \setminus \{C\}$ is satisfiable.

Lemma 14 (Semi-Relevance Test). Given a set of clauses $N$, and a clause $C \in N$, $C$ is semi-relevant if and only if $(N \setminus \{C\}, \{C\}) \Rightarrow_{RES}^* (N \setminus \{C\}, S \cup \{\bot\})$.

Proof. If $(N \setminus \{C\}, \{C\}) \Rightarrow_{RES}^* (N \setminus \{C\}, S \cup \{\bot\})$ then we have found a refutation containing $C$. On the other hand, by Theorem $11$, Lemma $7$, and Corollary $5$, if there is a refutation containing $C$, then there is also an SOS refutation with SOS $\{C\}$.
An immediate consequence of the above test and completeness of resolution for first-order logic is the following corollary.

**Corollary 15 (Complexity of the Semi-Relevance Test).** Testing semi-relevance in first-order logic is semi-decidable. It is decidable for all fragments where resolution constitutes a decision procedure.

Fragments where our semi-relevance test is guaranteed to terminate are for example first-order fragments enjoying the bounded model property, such as the Bernays-Schoenfinkel fragment \([3]\).

5 Conclusion

We have extended and sharpened the original completeness result for SOS resolution \([15]\), Theorem \([11]\). The generalized SOS completeness result can actually be used to effectively test clauses for semi-relevance in case resolution constitutes a decision procedure for the respective clause set. This is for example the case for all fragments enjoying the bounded model property, such as the Bernays-Schoenfinkel fragment \([3]\). In general, our approach yields a semi-decision procedure for semi-relevance.

Our proof is based on deductions having an a priori tree structure. However, this is not a principle restriction. It just simplifies the transformation introduced in Section \([2]\). Renamings have only to be considered on input clauses. In a setting where proofs forming directed acyclic graphs are considered, renamings have to be carried all over a deduction, adding further technicalities to our transformation.

It is well-known that changing the ordering of resolution steps in a resolution deduction may exponentially increase or exponentially decrease the length of the deduction. Therefore, our transformation of a deduction into an SOS deduction may also yield an exponential growth in the length of the deduction. It may also be the other way round if, e.g., subsumption is added to the transformation. It is also not difficult to find examples where the transformation of Section \([2]\) introduces redundant clauses. Recall that we have not made any assumption with respect to redundancy on deductions. So an open question is whether corresponding results hold on non-redundant deductions and what they actually mean for a respective notion of relevance.

An open problem is the question whether a test for semi-relevance can be established with more restricted resolution calculi such as ordered resolution. In general, the SOS strategy is not complete with ordered resolution. However, it is complete with respect to a clause set saturated by ordered resolution. The technical obstacle here is that a saturated clause set may already contain the empty clause, because for our generalized completeness result and the respective relationship to semi-relevance, the set \(N\) may still be unsatisfiable without the clause \(C\) to be tested for semi-relevance.

**Acknowledgments:** This work was funded by DFG grant 389792660 as part of TRR 248. We thank our reviewers for their valuable comments.
References

1. Baader, F., Peñaloza, R.: Axiom pinpointing in general tableaux. J. Log. Comput. 20(1), 5–34 (2010)
2. Bachmair, L., Ganzinger, H.: Rewrite-based equational theorem proving with selection and simplification. Journal of Logic and Computation 4(3), 217–247 (1994), revised version of Max-Planck-Institut für Informatik technical report, MPI-I-91-208, 1991
3. Bernays, P., Schönfinkel, M.: Zum entscheidungsproblem der mathematischen logik. Mathematische Annalen 99, 342–372 (1928)
4. Bourgaux, C., Ozaki, A., Peñaloza, R., Predoiu, L.: Provenance for the description logic elhr. In: Bessiere, C. (ed.) Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI 2020. pp. 1862–1869. ijcai.org (2020)
5. Eiter, T., Gottlob, G.: The complexity of logic-based abduction. Journal of the ACM 42(1), 3–42 (1995)
6. Fetzer, C., Weidenbach, C., Wischnewski, P.: Compliance, functional safety and fault detection by formal methods. In: Margaria, T., Steffen, B. (eds.) Leveraging Applications of Formal Methods, Verification and Validation: Discussion, Dissemination, Applications - 7th International Symposium, ISO LA 2016, Imperial, Corfu, Greece, October 10-14, 2016. Proceedings, Part II. Lecture Notes in Computer Science, vol. 9953, pp. 626–632 (2016)
7. Haifani, F., Koopmann, P., Tourret, S., Weidenbach, C.: On a notion of relevance. In: Borgwardt, S., Meyer, T. (eds.) Proceedings of the 33rd International Workshop on Description Logics (DL 2020) co-located with the 17th International Conference on Principles of Knowledge Representation and Reasoning (KR 2020), Online Event [Rhodes, Greece], September 12th to 14th, 2020. CEUR Workshop Proceedings, vol. 2663. CEUR-WS.org (2020)
8. Kalyanpur, A., Parsia, B., Horridge, M., Sirin, E.: Finding all justifications of OWL DL entailments. In: Aberer, K., Choi, K., Noy, N.F., Allemang, D., Lee, K., Nixon, L.J.B., Golbeck, J., Mika, P., Maynard, D., Mizoguchi, R., Schreiber, G., Cudré-Mauroux, P. (eds.) The Semantic Web, 6th International Semantic Web Conference, 2nd Asian Semantic Web Conference, ISWC 2007 + ASWC 2007, Busan, Korea, November 11-15, 2007. Lecture Notes in Computer Science, vol. 4825, pp. 267–280. Springer (2007)
9. Kleine Büning, H., Kullmann, O.: Minimal unsatisfiability and autarkies. In: Biere, A., Heule, M., van Maaren, H., Walsh, T. (eds.) Handbook of Satisfiability, Frontiers in Artificial Intelligence and Applications, vol. 185, pp. 339–401. IOS Press (2009)
10. Kullmann, O.: Investigations on autark assignments. Discret. Appl. Math. 107(1-3), 99–137 (2000)
11. Lee, C.T.: A Completeness Theorem and a Computer Program for Finding Theorems Derivable from Given Axioms. Phd thesis, University of Berkeley, California, Department of Electrical Engineering (1967)
12. Lev-Ami, T., Weidenbach, C., Reps, T.W., Sagiv, M.: Labelled clauses. In: Pfenning, F. (ed.) Automated Deduction - CADE-21, 21st International Conference on Automated Deduction, Bremen, Germany, July 17-20, 2007. Proceedings. LNCS, vol. 4603, pp. 311–327. Springer (2007)
13. Nienhuys-Cheng, S., de Wolf, R.: The equivalence of the subsumption theorem and the refutation-completeness for unconstrained resolution. In: Kachanavsat, K., Lévy, J. (eds.) Algorithms, Concurrency and Knowledge: 1995 Asian Computing Science
14. Robinson, J.A.: A machine-oriented logic based on the resolution principle. Journal of the ACM 12(1), 23–41 (January 1965)
15. Robinson, J.A., Voronkov, A. (eds.): Handbook of Automated Reasoning (in 2 volumes). Elsevier and MIT Press (2001)
16. Schlobach, S., Cornet, R.: Non-standard reasoning services for the debugging of description logic terminologies. In: Gottlob, G., Walsh, T. (eds.) IJCAI-03, Proceedings of the Eighteenth International Joint Conference on Artificial Intelligence, Acapulco, Mexico, August 9-15, 2003. pp. 355–362. Morgan Kaufmann (2003)
17. Walter, R., Felfernig, A., Küchlin, W.: Constraint-based and sat-based diagnosis of automotive configuration problems. J. Intell. Inf. Syst. 49(1), 87–118 (2017)
18. Wos, L., Robinson, G., Carson, D.: Efficiency and completeness of the set of support strategy in theorem proving. Journal of the ACM 12(4), 536–541 (1965)