Implicational Completeness of Signed Resolution

Christian G. Fermüller*
Technische Universität Wien

1 Implicational completeness - a neglected topic

Every serious computer scientist and logician knows that resolution is complete for first-order clause logic. By this, of course, one means that the empty clause (representing contradiction) is derivable by resolution from every unsatisfiable set of clauses $\mathcal{S}$. However, there is another – less well known – concept of completeness for clause logic, that is often referred to as "Lee's Theorem" (see, e.g., [8]): Char-tung Lee's dissertation [7] focused on an interesting observation that (in a corrected version and more adequate terminology) can be stated as follows:

Theorem 1.1 (Lee) *Let $\mathcal{S}$ be a set of clauses. For every non-tautological clause C that is logically implied by $\mathcal{S}$ there is clause D, derivable by resolution from $\mathcal{S}$, such that D subsumes C.*

Observe that this theorem amounts to a *strengthening* of refutational completeness of resolution: If $\mathcal{S}$ is unsatisfiable then it implies every clause; but the only clause that subsumes every clause (including the empty clause) is the empty clause, which therefore must be derivable by resolution from $\mathcal{S}$ according to the theorem.

At least from a logical point of view, Lee's "positive" completeness result is as interesting as refutational completeness. Nevertheless this classic result – which we prefer to call *implicational completeness* of resolution – is not even mentioned in most textbooks and survey articles on automated deduction. The main reason for this is probably the conception that implicational completeness, in contrast to refutational completeness, is of no practical significance. Moreover, it fails for all important *refinements* of Robinson's original resolution calculus. In addition, Lee's proof [7] is presented in an unsatisfactory manner (to say the least). A fourth reason for the widespread neglect of implicational completeness might be the fact that Lee (and others at that time) did not distinguish between *implication* and *subsumption* of clauses. However, nowadays, it is well known that the first relation between clauses is undecidable [10], whereas sophisticated and efficient algorithms for testing the latter one are at the core of virtually all successful resolution theorem provers (see, e.g., [4]). With hindsight, this is decisive for the significance of Lee's Theorem.

We will provide a new and independent proof of implicational completeness in a much more general setting, namely *signed resolution*. An additional motivation is that this result is needed for an interesting application: computing optimal rules for the handling of

*Institut für Computersprachen 185.2, Resselgass 3/3, A-1040 Vienna, email: ChrisF@logic.at

quantifiers in many-valued logics (see [9]). In fact, we provide a self-contained presentation of signed resolution (compare [5, 2]).

Readers mainly interested in classical logic are reminded that classical clause logic is just the simplest case of signed resolution. Even for this special case our proof is new and independent from the (rather intricate) ones presented in [7] and [8].

2 Signed clause logic

Atomic formulae – or: *atoms* – are build up from predicate, function and variable symbols as usual. (Constants are considered as function symbols of arity 0.) By the *Herbrand base* (corresponding to some signature) we mean the set of all *ground* atoms; i.e., atoms that do not contain variable symbols. We consider the reader to be familiar with other standard notions, like *substitution*, *most general unifier (mgu)* etc.

Let W be a fixed finite set; here always considered as the set of truth values. A *literal* (over W) is an expression $S\colon P$, where P is an atom and $S \subseteq W$. A *(signed) clause* is a finite set of literals.[1]

An *assignment* associates truth values (i.e., elements of W) with atoms. A *complete assignment to a set of atoms K* is defined as a set of literals $\{\{\psi(P)\}\colon P \mid P \in K\}$, where ψ is a total function from K to W. An *(Herbrand-)interpretation* is a complete assignment of the Herbrand base.

For any set of atoms K the corresponding *literal set* $\Lambda(K)$ is the set $\{V\colon A \mid A \in K, V \subseteq W, V \neq \emptyset\}$.

To assist concise statements about the relation between arbitrary sets of literals we use the following notation:

For a set of literals C let $\widehat{C}$ be the equivalent set that consists of singleton-as-sign literals only. More exactly, $\widehat{C} = \{\{v\}\colon A \mid S\colon A \in C, v \in S\}$. We say that C is *contained* in another set of literals D if $\widehat{C} \subseteq \widehat{D}$.

An interpretation I *satisfies* a clause set $\mathcal{S}$ iff for all ground instances C' of each $C \in \mathcal{S}$: $\widehat{C'} \cap I \neq \emptyset$. I is called an *H-model* of $\mathcal{S}$. S is *(H-)unsatisfiable* if it has no H-model. Since an analogue of Herbrand's theorem holds for signed clause logic (see, e.g., [1, 2]) we can restrict our attention to H-models.

The significance of this notions lies in the fact that formulae of any first-order finite-valued logic can effectively be translated to finite sets of signed clauses in such a way that the clause set is unsatisfiable iff the original formula is valid in the source logic. (See, e.g., [6, 2] for a detailed presentation of this fact.)

3 Signed resolution

The conclusion of the following inference rule:

$$\frac{\{S\colon P\} \cup C_1 \qquad \{R\colon Q\} \cup C_2}{(\{S \cap R\colon P\} \cup C_1 \cup C_2)\sigma} \text{ binary resolution}$$

is called a *binary resolvent* of the variable disjoint *parent clauses* $\{S\colon P\} \cup C_1$ and $\{R\colon Q\} \cup C_2$, if $S \neq R$ and σ is an mgu of the atoms P and Q.

[1] In classical clause logic we have $W = \{\mathbf{true}, \mathbf{false}\}$. Literals $\mathbf{true}\colon P$ and $\mathbf{false}\colon P$ are traditionally denoted as simply as P and $\neg P$, respectively.

Like in the classical case we need a *factorization rule* to obtain a refutationally complete calculus:

$$\frac{C}{C\sigma} \text{ factorization}$$

where σ is an mgu of a subset of C. $C\sigma$ is called a *factor* of C.

The combination of factorization and binary resolution does not yet guarantee that the empty clause can be derived from all unsatisfiable sets of clauses. We also have to remove literals with empty signs by the following *simplification rule*:[2]

$$\frac{C \cup \{\emptyset \colon P\}}{C} \text{ simplification}$$

C is called a *simplification* of C' if it results from C' by removing all literals with empty sign. (I.e., by applying the simplification rule to C' as often as possible.)

The *merging rule* unites literals that share the same atom. It is not needed for completeness but helps to reduce the search space and to simplify the completeness proof.[3]

$$\frac{\{S_1 \colon P\} \cup \ldots \cup \{S_n \colon P\} \cup C}{\{S_1 \cup \ldots \cup S_n \colon P\} \cup C} \text{ merging}$$

C is called a *normal form* or *normalized* version of C' if it results from C' by applying the simplification rules and the merging rule to C' as often as possible. I.e., all literals with empty signs are removed and all different literals in C have different atoms.

One can combine factoring, simplification, merging, and binary resolution into a single resolution rule. This corresponds to a particular strategy for the application of these rules.

The following alternative version of signed resolution can be considered as a combination of a series of binary resolution and simplification steps into one "macro inference step", called *hyperresolution* in [5].

$$\frac{\{S_1 \colon P_1\} \cup C_1 \qquad \ldots \qquad \{S_n \colon P_n\} \cup C_n}{(C_1 \cup \ldots \cup C_n)\sigma} \text{ hyperresolution}$$

where $S_1 \cap \ldots \cap S_n = \emptyset$ and σ is the mgu of the atoms P_i $(1 \leq i \leq n)$. The conclusion is called a *hyperresolvent.*

It is useful to consider resolution as a set operator (mapping sets of clauses into sets of clauses).

For a set of clauses S let $\mathcal{R}_b(S)$ be the set of all binary resolvents of (variable renamed) normalized factors of clauses in S. The transitive and reflexive closure of the set operator $\mathcal{R}_b$ is denoted by $\mathcal{R}_b^*$.

Similarly, we define $\mathcal{R}_m(S)$ as the set of all hyperresolvents of (variable renamed) normalized factors of clauses in S. $\mathcal{R}_m^*$ denotes the transitive and reflexive closure of $\mathcal{R}_m$.

A resolution operator $\mathcal{R}$ is *refutationally complete* if, for all clause sets S, S unsatisfiable implies $\{\} \in \mathcal{R}^*(S)$.

A resolution operator $\mathcal{R}$ is called *implicationally complete* if, for all clause sets S and clauses C, either C is a tautology or C is subsumed by some $C' \in \mathcal{R}^*(S)$ whenever S implies C.

[2]Alternatively, one can dispose with the simplification rule by defining a clause to be empty if all literals have empty sets as signs.

[3]The merging rule is needed for completeness if clauses are not treated modulo idempotency of disjunction (e.g., as multisets as opposed to sets).

Observe that hyperresolution does *not* enjoy implicational completeness: Consider, e.g., the propositional clauses

$$\{\{u,v\}\colon A\} \quad \text{and} \quad \{\{u,w\}\colon A\},$$

where u, v, w are pairwise different truth values. The hyperresolution rule is not applicable. The (non-tautological) clause $\{\{u\}\colon A\}$ is implied by $\{\{\{u,v\}\colon A\}, \{\{u,w\}\colon A\}\}$, without being subsumed by one of its members.

4 Implication and subsumption

For a clause $C = \{S_1\colon P_1, \ldots S_N\colon P_n\}$ let

$$[\neg C] = \{\{W - S_1\colon P_1\gamma\}, \ldots \{W - S_n\colon P_n\gamma\}\},$$

where γ is a substitution that replaces each variable in C by a new constant. (W is the set of all truth values.)

Proposition 4.1 *For every clause C and interpretation I*[4]*: I is a model of $[\neg C]$ iff I does not satisfy C.*

Proof. Follows from the definition of $[\neg C]$. ∎

A clause C *subsumes* a clause D if some instance of C is contained in D; more formally: if $\widehat{C\theta} \subseteq \widehat{D}$, for some substitution θ. A set of clauses $\mathcal{S}$ *implies* a clause C if all models of $\mathcal{S}$ satisfy C. We state some simple facts about implication of clauses and subsumption.

Proposition 4.2 *Let C and D be clauses. If C subsumes D then $\{C\}$ implies D.*

Proof. Follows from the definitions of subsumption and implication, respectively. ∎

Observe that the converse of Proposition 4.2 does not hold. E.g., $\{\{u\}\colon P(x), \{v\}\colon P(f(x)\}$ implies but does not subsume $\{\{u\}\colon P(x), \{v\}\colon P(f(f(x))\}$ if $u \neq v$. Whereas the problem to decide whether a clause C subsumes a clause D is NP-complete (see [3], it is undecidable whether $\{C\}$ implies D, in general as proved in [10].

Proposition 4.3 *Let S be a clause set and C be a non-tautological clause. S implies C iff $S \cup [\neg C]$ is unsatisfiable.*

Proof. Follows from Proposition 4.1 and the definition of implication. ∎

Lemma 4.1 *Let C and D be non-tautological clauses. C subsumes D iff there exists a ground substitution θ s.t. $\{C\theta\} \cup [\neg D]$ is unsatisfiable.*

Proof. $\Rightarrow$: Suppose $\widehat{C\sigma} \subseteq \widehat{D}$. Then also $\widehat{C\sigma\gamma} \subseteq \widehat{D\gamma}$, where γ is the substitution replacing every variable by a new constant in $[\neg D]$. This implies that for each literal $V\colon A \in \widehat{C\sigma\gamma}$, there is a clause of form $\{V'\colon A\} \in [\neg D]$ such that $V \cap V' = \emptyset$. This means that $\{C\sigma\gamma\} \cup [\neg D]$ is unsatisfiable.

$\Leftarrow$: Suppose $\{C\theta\} \cup [\neg D]$ is unsatisfiable, where $C\theta$ is ground. Since D is non-tautological, $[\neg D]$ is satisfiable. Therefore, for each literal $\{v\}\colon A \in \widehat{C\theta}$ there has to exist a clause $\{S\colon A\} \in [\neg D]$ s.t. $v \notin S$. This implies $\widehat{C\theta} \subseteq \widehat{D}$. In other words: C subsumes D. ∎

[4] Of course, the Herbrand universe has to include also the new constants occurring in $[\neg C]$.

5 Semantic trees for signed clause logic

Our completeness proof is based on the concept of semantic trees. It differs from the proofs in [1] and [6]; but generalizes the completeness proof in [2] for singletons-as-signs resolution to (unrestricted) signed resolution.

As usual in automated deduction, we consider a tree as growing downwards; i.e. the *root* is the top node of a tree. A node or edge α *is above* a node or edge β if α is part of the path (considered as alternating sequence of nodes and edges) connecting β with the root. A *branch* of T is a path that starts with the root and either is infinite or else ends in a leaf node of T.

Let W be a finite set of truth values and K be a set of ground atoms. For any subset Δ of the literal set $\Lambda(K)$ of K we say that Δ *omits* the assignment A_K to K if $\Delta \cap A_K = \emptyset$. A finitely branching tree T is a *semantic tree for* K if finite, non-empty subsets of $\Lambda(K)$ label the edges of T in the following way:

(1) The set of the sets of literals labeling all edges leaving one node is an H-unsatisfiable set of clauses.

(2) For each branch of T the union of the sets of literals labeling the edges of the branch omits exactly one complete assignment A_K to K. For short, we say that *the branch omits* A_K as well as any interpretation containing A_K.

(3) For each complete assignment A_K to K there is a branch of T s.t. this branch omits A_K.

The union of all sets of literals labeling the edges of the path from the root down to some node α of T forms the *refutation set* of α.

For a set of clauses S any semantic tree T for $A(S)$ represents an exhaustive survey of all possible H-interpretations. Each branch omits exactly one H-interpretation and each H-interpretation is omitted by at least one branch.

A clause C *fails* at a node α of a semantic tree T if some ground instance of C is contained in the refutation set of that node. A node α is a *failure node* for a clause set S if some clause of S fails at α but no clause in S fails at a node above α. A node is called an *inference node* if all of its successor nodes are failure nodes. T is *closed* for $\mathcal{S}$ if there is a failure node for S on every branch of T.

Theorem 5.1 *A set of clauses S is unsatisfiable iff there is a finite subset $K \subseteq A(S)$ s.t. every semantic tree for K is closed for S.*

Proof. $\Rightarrow$: Let T be a semantic tree for $A(S)$, the Herbrand base of S. By definition of a semantic tree, any branch B of T omits exactly one complete assignment to $A(S)$, which extends to an H-interpretation $\mathcal{M}$ of S. If S is unsatisfiable then $\mathcal{M}$ does not satisfy all clauses in S. This means that there is some ground instance C' of a clause C in S s.t. $\widehat{C'} \cap \mathcal{M} = \emptyset$. But since B omits only the literals of $\Lambda(A(S))$ that are true in $\mathcal{M}$ this implies that the union of labels of the edges of B contains C'; i.e., C' is contained in the refutation set of some node of B. We have thus proved that every branch of T contains a failure node for some clause of S. In other words, T is closed for S. Moreover, by König's Lemma, the number of nodes in T that are situated above a failure node is finite. But this implies that for each unsatisfiable set of clauses S there is a finite unsatisfiable set S' of ground instances of clauses of S. Since any semantic tree that is closed for S' is also closed for S it is sufficient to base the tree on a finite subset of $A(S)$: the set K of ground atoms occurring in S'. Observe that we have not imposed any restriction on the form of the tree. Thus every semantic tree for K is closed for S.

$\Leftarrow$: Let T be a closed semantic tree for a finite $K \subseteq A(S)$. Suppose $\mathcal{M}$ is an H-model of S; i.e. for all ground instances C' of $C \in S$ we have $\mathcal{M} \cap \widehat{C'} \neq \emptyset$. By definition of a semantic tree, $\mathcal{M}$ is omitted by some branch B of T. Since T is closed, some clause $C \in S$ fails at a node α of B. That means that some ground instance C' of C is contained in the refutation set of α. Therefore $\mathcal{M} \cap \widehat{C'} \neq \emptyset$ implies that $\mathcal{M}$ contains some literal that also occurs in some refuation set of a node on B. But this contradicts the assumption that B omits $\mathcal{M}$. Therefore S is unsatisfiable. ■

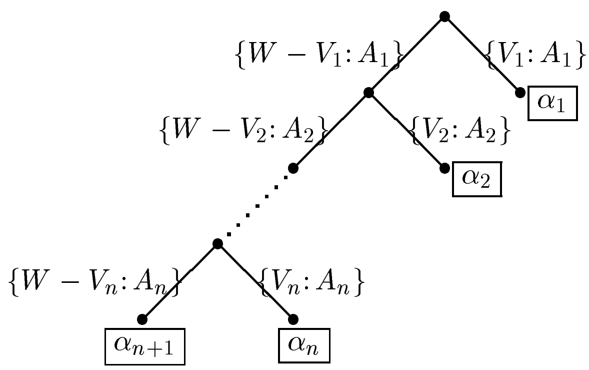
Theorem 5.1 is the basis for refutional completeness proofs for many different versions and refinements of signed resolution (see [2]). Our task here is to show that it can be used to prove implicational completeness as well.

6 Implicational completeness

Theorem 6.1 *$\mathcal{R}_b$ is implicationally complete. More precisely, if C is a non-tautological clause that is implied by a set of clause S then there exists a $D \in \mathcal{R}_b^*(S)$ s.t. D subsumes C.*

Proof. By Propositon 4.3 $S \cup [\neg C]$ is unsatisfiable. Hence, by Theorem 5.1 there is finite subset K of $A(S \cup [\neg C])$ s.t. every semantic tree for K is closed for $S \cup [\neg C]$.

Let $[\neg C] = \{\{V_1\colon A_1\}, \ldots, \{V_n\colon A_n\}\}$ and W be the set of all truth values. Since C is non-tautological $W - V_i$ is not empty. Without loss of generality we may assume C to be normalized; i.e., $A_i \neq A_j$ if $i \neq j$. We choose a semantic tree T for K that starts with the following subtree:



The subtrees of T rooted in the nodes $\alpha_1, \ldots, \alpha_n$, respectively, are arbitrary (since these nodes obviously are failure nodes).

For the construction of the subtree T_{n+1} of T rooted in α_{n+1} we have to take care that it does not contain a failure node for any clause in $[\neg C]$. This can be achieved as follows. Let $V_1^1, \ldots, V_1^k$ be the subsets of V_1 that contain all but one element of V_1. (If V_1 is a singleton simply skip this part of the construction of T.) Attach k successor nodes $\beta_1, \ldots, \beta_k$ to α_{n+1}, the edges to which are labeled by $\{V_1^1\colon A_1\}$, …, $\{V_1^k\colon A_1\}$, respectively. Clearly, the refutation set of β_i $(1 \leq i \leq k)$ omits exactly one assignment to the atom A_1. By proceeding in the same way for $A_2, \ldots A_n$ we arrive at a partial semantic tree T_C, each branch of which omits exactly one assignment to the atoms occurring in $[\neg C]$. Thus no literals signing atoms of $[\neg C]$ will have to occur below T_C. Therefore we can assume that the only failure nodes in T of clauses in $[\neg C]$ are $\alpha_1, \ldots, \alpha_n$. In other words: all failure nodes in T_{n+1} are failure nodes for clauses in S.

The only restriction (in addition to the requirement that T is a semantic tree for K) that we pose on the structure of T below T_C is that the literals labeling edges directly connected to a common node all contain the same atom. This way the following statement is easily seen to follow from condition (1) of the definition of a semantic tree.

(R) Let α be an inference node in T. Let $C_1, \ldots C_n$ be the clauses failing at its successor nodes $\beta_1, \ldots \beta_n$, respectively. Then some resolvent $D \in \mathcal{R}_b^*(\{C_1, \ldots C_n\})$ fails at α.

Since T is closed for $S \cup [\neg C]$ it must contain at least one inference node. Therefore, by iteratively adding resolvents to $S \cup [\neg C]$ and applying **(R)**, we must eventually derive a clause D that fails at the node α_{n+1}. Since T_{n+1} contains no failure nodes for clauses in $[\neg C]$ we conclude that $D \in \mathcal{R}_b^*(S)$. By Theorem 5.1 it follows that$\{D\theta\} \cup [\neg C]$ is unsatisfiable, where θ is a ground substitution such that $D\theta$ is contained in the refutation set of node α_{n+1}. By Lemma 4.1 it follows that D subsumes C. ∎

References

[1] M. Baaz. Automatisches Beweisen für endlichwertige Logiken. In *Jahrbuch 1989 der Kurt Gödel-Gesellschaft*, pages 105–107. Kurt Gödel Society, 1989.

[2] M. Baaz and C. G. Fermüller. Resolution-based theorem proving for many-valued logics. *J. Symbolic Computation*, 19:353–391, 1995.

[3] M.S. Garey and D.S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness.* Freeman, San Francisco, 1979.

[4] G. Gottlob and A. Leitsch. On the efficiency of subsumbtion algorithms. *Journal of the ACM*, 32(2):280–295, 1985.

[5] R. Hähnle. *Automated Deduction in Multiple-valued Logics.* Clarendon Press, Oxford, 1993.

[6] R. Hähnle. Short conjunctive normal forms in finitely-valued logics. *Journal of Logic and Computation*, 4(6):905–927, 1994.

[7] R.C.T. Lee. *A completeness theorem and a computer program for finding theorems derivable from given axioms.* Ph.D. Thesis, University of California, Berkely, 1967.

[8] A. Leitsch. *The Resolution Calculus.* Springer, Berlin, Heidelberg, New York, 1997.

[9] G. Salzer. Optimal axiomatizations for multiple-valued operators and quantifiers based on semi-lattices. In *13th Int. Conf. on Automated Deduction (CADE'96)*, LNCS (LNAI). Springer, 1996.

[10] M. Schmidt-Schauss. Implication of clauses is undecidable. *Theoretical Computer Science*, 59:287–296, 1988.