Proof. Let C_1 subsume C_2 with substitution σ Subsumption is in NP because the size of σ is bounded by the size of C_2 and the subset relation can be checked in time at most quadratic in the size of C_1 and C_2.

Propositional SAT can be reduced as follows. Assume a 3-SAT clause set N. Consider a 3-place predicate R and a unary function g and a mapping from propositional variables P to first order variables x_P. $\square$

Lemma 3.13.7 (Lifting)**.** Let $D \vee L$ and $C \vee L'$ be variable-disjoint clauses and σ a grounding substitution for $C \vee L$ and $D \vee L'$. If there is a superposition left inference

$$(N \uplus \{(D \vee L)\sigma, (C \vee L')\sigma\}) \Rightarrow_{\text{SUP}} (N \cup \{(D \vee L)\sigma, (C \vee L')\sigma\} \cup \{D\sigma \vee C\sigma\})$$

and if $\text{sel}((D \vee L)\sigma) = \text{sel}((D \vee L))\sigma$, $\text{sel}((C \vee L')\sigma) = \text{sel}((C \vee L'))\sigma$, then there exists a mgu τ such that

$$(N \uplus \{D \vee L, C \vee L'\}) \Rightarrow_{\text{SUP}} (N \cup \{D \vee L, C \vee L'\} \cup \{(D \vee C)\tau\}).$$

Let $C \vee L \vee L'$ be a clause and σ a grounding substitution for $C \vee L \vee L'$. If there is a factoring inference

$$(N \uplus \{(C \vee L \vee L')\sigma\}) \Rightarrow_{\text{SUP}} (N \cup \{(C \vee L \vee L')\sigma\} \cup \{(C \vee L)\sigma\})$$

and if $\text{sel}((C \vee L \vee L')\sigma) = \text{sel}((C \vee L \vee L'))\sigma$, then there exists a mgu τ such that

$$(N \uplus \{C \vee L \vee L'\}) \Rightarrow_{\text{SUP}} (N \cup \{C \vee L \vee L'\} \cup \{(C \vee L)\tau\})$$

Note that in the above lemma the clause $D\sigma \vee C\sigma$ is an instance of the clause $(D \vee C)\tau$. The reduction rules cannot be lifted in the same way as the following example shows.

Example 3.13.8 (First-Order Reductions are not Liftable)**.** Consider the two clauses $P(x) \vee Q(x)$, $P(g(y))$ and grounding substitution $\{x \mapsto g(a), y \mapsto a\}$. Then $P(g(y))\sigma$ subsumes $(P(x) \vee Q(x))\sigma$ but $P(g(y))$ does not subsume $P(x) \vee Q(x)$. For all other reduction rules similar examples can be constructed.

Lemma 3.13.9 (Soundness and Completeness)**.** First-Order Superposition is sound and complete.

Proof. Soundness is obvious. For completeness, Theorem 3.12.12 proves the ground case. Now by applying Lemma 3.13.7 to this proof it can be lifted to the first-order level, as argued in the following.

Let N be a an unsatisfiable set of first-order clauses. By Theorem 3.5.5 and Lemma 3.6.10 there exist a finite unsatisfiable set N' of ground instances from clauses from N such that for each clause $C\sigma \in N'$ there is a clause $C \in N$. Now ground superposition is complete, Theorem 3.12.12, so there exists a derivation of the empty clause by ground superposition from N': $N' = N'_0 \Rightarrow_{\text{SUP}} \ldots \Rightarrow_{\text{SUP}} N'_k$ and $\bot \in N'_k$. Now by an inductive argument on the length of the derivation k this derivation can be lifted to the first-order level. The invariant is: for any ground clause $C\sigma \in N'_i$ used in the ground proof, there is a clause $C \in N_i$ on the first-order level. The induction base holds for N and N' by construction. For the induction step Lemma 3.13.7 delivers the result. □

There are questions left open by Lemma 3.13.9. It just says that a ground refutation can be lifted to a first-order refutation. But what about abstract redundancy, Definition 3.13.3? Can first-order redundant clauses be deleted without harming completeness? And what about the ground model operator with respect to clause sets N saturated on the first-order level. Is in this case $\operatorname{grd}(\Sigma, N)_{\mathcal{I}}$ a model? The next two lemmas answer these questions positively.

Lemma 3.13.10 (Redundant Clauses are Obsolete)**.** If a clause set N is unsatisfiable, then there is a derivation $N \Rightarrow^*_{\text{SUP}} N'$ such that $\bot \in N'$ and no clause in the derivation of $\bot$ is redundant.

Proof. If N is unsatisfiable then there is a ground superposition refutation of $\operatorname{grd}(\Sigma, N)$ such that no ground clause in the refutation is redundant. Now according to Lemma 3.13.9 this proof can be lifted to the first-order level. Now assume some clause C in the first-order proof is redundant that is the lifting of some clause $C\sigma$ from the ground proof with respect to a grounding substitution σ. The clause C is redundant by Definition 3.13.3 if all its ground instances are, in particular, $C\sigma$. But this contradicts the fact that the lifted ground proof does not contain redundant clauses. □

Lemma 3.13.11 (Model Property)**.** If N is a saturated clause set and $\bot \notin N$ then $\operatorname{grd}(\Sigma, N)_{\mathcal{I}} \models N$.

Proof. As usual we assume that selection on the ground and respective non-ground clauses is identical. Assume $\operatorname{grd}(\Sigma, N)_{\mathcal{I}} \not\models N$. Then there is a minimal ground clause $C\sigma$, $C \neq \bot$, $C \in N$ such that $\operatorname{grd}(\Sigma, N)_{\mathcal{I}} \not\models C\sigma$. Note that $C\sigma$ is not redundant as for otherwise $\operatorname{grd}(\Sigma, N)_{\mathcal{I}} \models C\sigma$. So $\operatorname{grd}(\Sigma, N)$ is not saturated. If $C\sigma$ is productive, i.e., $C\sigma = (C' \vee L)\sigma$ such that L is positive, $L\sigma$ strictly maximal in $(C' \vee L)\sigma$ then $L\sigma \in \operatorname{grd}(\Sigma, N)_{\mathcal{I}}$ and hence $\operatorname{grd}(\Sigma, N)_{\mathcal{I}} \models C\sigma$ contradicting $\operatorname{grd}(\Sigma, N)_{\mathcal{I}} \not\models C\sigma$.

If $C\sigma = (C' \vee L \vee L')\sigma$ such that L is positive, $L\sigma$ maximal in $(C' \vee L \vee L')\sigma$ then, because N is saturated, there is a clause $(C' \vee L)\tau \in N$ such that $(C' \vee L)\tau\sigma = (C' \vee L)\sigma$. Now $(C' \vee L)\tau$ is not redundant, $\operatorname{grd}(\Sigma, N)_{\mathcal{I}} \not\models (C' \vee L)\tau$, contradicting the minimal choice of $C\sigma$.

If $C\sigma = (C' \vee L)\sigma$ such that L is selected, or negative and maximal then there is a clause $(D' \vee L') \in N$ and grounding substitution ρ, such that $L'\rho$ is a strictly maximal positive literal in $(D' \vee L')\rho$, $L'\rho \in \operatorname{grd}(\Sigma, N)_{\mathcal{I}}$ and $L'\rho = \neg L\sigma$. Again, since N is saturated, there is variable disjoint clause $(C' \vee D')\tau \in N$ for some unifier τ, $(C' \vee D')\tau\sigma\rho \prec C\sigma$, and $\operatorname{grd}(\Sigma, N)_{\mathcal{I}} \not\models (C' \vee D')\tau\sigma\rho$ contradicting the minimal choice of $C\sigma$. □

Dynamic stuff: a clause C is called *persistent* in a derivation $N \rightarrow^*_{\text{SUP}} N'$ if there is some i such that $C \in N_i$ for $N \rightarrow^i_{\text{SUP}} N_i$ and for all $j > i$, $N \rightarrow^j_{\text{SUP}} N_j$ then $C \in N_j$. A derivation $N \rightarrow^*_{\text{SUP}} N'$ is called *fair* if any two persistent clauses C, D and any superposition inference C' out of the two clauses, there is an index j such with $N \rightarrow^j_{\text{SUP}} N_j \rightarrow^*_{\text{SUP}} N'$ such that $C' \in N_j$.

Definition 3.13.12 (Persistent Clause)**.** Let $N_0 \Rightarrow_{\text{SUP}} N_1 \Rightarrow_{\text{SUP}} \ldots$ be a, possibly infinite, superposition derivation. A clause C is called *persistent* in this derivation if $C \in N_i$ for some i and for all $j > i$ also $C \in N_j$.

Definition 3.13.13 (Fair Derivation)**.** A derivation $N_0 \Rightarrow_{\text{SUP}} N_1 \Rightarrow_{\text{SUP}} \ldots$ is called *fair* if for any persistent clause $C \in N_i$ where factoring is applicable to C, there is a j such that the factor of $C' \in N_j$ or $\bot \in N_j$. If $\{C, D\} \subseteq N_i$ are persistent clauses such that superposition left is applicable to C, D then the superposition left result is also in N_j for some j or $\bot \in N_j$.

Theorem 3.13.14 (Dynamic Superposition Completeness)**.** If N is unsatisfiable and $N = N_0 \Rightarrow_{\text{SUP}} N_1 \Rightarrow_{\text{SUP}} \ldots$ is a fair derivation, then there is $\bot \in N_j$ for some j.

Proof. If N is unsatisfiable, then by Lemma 3.13.9 there is a derivation of $\bot$ by superposition. Furthermore, no clause contributing to the derivation of $\bot$ is redundant (Lemma 3.13.10). So all clauses in the derivation of $\bot$ are persistent. The derivation $N_0 \Rightarrow_{\text{SUP}} N_1 \Rightarrow_{\text{SUP}} \ldots$ is fair, hence $\bot \in N_j$ for some j. □

Lemma 3.13.15. Let $\text{red}(N)$ be all clauses that are redundant with respect to the clauses in N and N, M be clause sets. Then

1. if $N \subseteq M$ then $\text{red}(N) \subseteq \text{red}(M)$
2. if $M \subseteq \text{red}(N)$ then $\text{red}(N) \subseteq \text{red}(N \setminus M)$

It follows that redundancy is preserved when, during a theorem proving process, new clauses are added (or derived) or redundant clauses are deleted. Furthermore, $\text{red}(N)$ may include clauses that are not in N.

3.14 Implementation

3.14.1 A First-Order Superposition Theorem Prover

So far: static view on completeness of resolution: Saturated sets are inconsistent if and only if they contain $\bot$. This chapter considers a dynamic view:

1. How to achieve saturated sets in practice?
2. The theorems and are the basis for the completeness proof of the prover *STP*.

Rules for Simplifications and Deletion

The following rules are employed for simplification of prover states N (there are more possibilities):

is obviously terminating, e.g., for an LPO with $f_R \succ g_a \succ g_b \succ c \succ d$, but not confluent, in general. Completing the system corresponds to searching for a PCP solution.

Although validity (unsatisfiability) checking in equational and first-order logic is undecidable, in general, there are meaningful subclasses such that these problems become decidable. The first class I want to look at is a "classics" the Bernays-Schönfinkel class.

Definition 3.15.3 (Bernays-Schönfinkel Fragment (BS))**.** A formula of the Bernays-Schönfikel fragment has the form $\exists\vec{x}.\forall\vec{y}.\phi$ such that ϕ is quantifier free and does not contain constant symbols nor function symbols.

Transforming a Bernays-Schönfinkel formula into CNF via $\Rightarrow_{\mathrm{ACNF}}$ also results in a set of clauses that only contains predicates, constant symbols, and universally quantified variables. Such clause sets are also called Bernays-Schönfinkel. The leading existential quantifiers $\exists\vec{x}$ of a BS formula $\exists\vec{x}\forall\vec{y}\phi$ are all Skolemized to Skolem constants. Therefore, a formula $\exists\vec{x}.\forall\vec{y}.\phi$ where ϕ contains constant symbols can be transformed into a BS formula by replacing the constant symbols with existentially quantified variables.

Theorem 3.15.4 (BS is decidable)**.** Unsatisfiability of a BS clause set is decidable.

Proof. The set of all ground terms of a BS clause set N is finite. By Herbrand's theorem, Theorem 3.5.5, a clause set is unsatisfiable iff $\mathrm{grd}(\Sigma, N)$ is. But $\mathrm{grd}(\Sigma, N)$ is finite, so its unsatisfiability is decidable by any propositional calculus. □

Interestingly, neither Resolution, nor Superposition terminate on a BS clause set, in general. Further refinements to these calculi are needed. Tableau terminates on BS clause sets. On the other hand, a clause set N is unsatisfiable iff the finite clause set $\mathrm{grd}(\Sigma, N)$ is unsatisfiable. After ground instantiation this can be decided by any propositional calculus. So if $\mathrm{grd}(\Sigma, N)$ is not "too large" this may be even an effective idea in practice. However $\mathrm{grd}(\Sigma, N)$ grows exponentially in the number of variables contained in clauses in N.

Deciding satisfiability of the BS fragment is NEXPTIME-complete. If the BS fragment is further restricted to Horn clauses, satisfiability of the BS fragment becomes DEXPTIME-complete. This is a common phaenomen for decidable logic clause fragments that if the general fragment is complete for the non-deterministic class, its Horn restriction is complete for the respective deterministic class. For propositional logic, satisfiability of arbitrary clauses is NP-complete, but satisfiability of Horn clauses is in P, actually it can be decided in linear time, see Proposition 2.14.7.

The fact that the BS fragment is NEXPTIME-complete has consequences for the size of models. Since NEXPTIME $\neq$ NP the size of BS models cannot be polynomial in the size of an input clause set. Actually, searching for formalisms that enable compact representations of BS models and efficient computations is an active area of research.

$\boxed{\text{C}}$ The class is expressive enough to encode many knowledge representation languages such as description logics or the ontology languages of the WWW consortium. Furthermore, it has the potential to deliver exponentially shorted refutations compared to reasoning on the ground level. Therefore, it has the potential to support progress in applications where currently propositional reasoning is preferred.

Example 3.15.5. Non-Termination of Superposition on BS Consider the clause set

$$N = \{\neg P(x,y) \lor P(x,z) \lor P(z,y),\ P(a,a)\}$$

In the first clause all three literals are incomparable with respect to any ordering. So there is at a least the superposition inference between the negative literal $\neg P(x,y)$ and $P(a,a)$ generating $P(a,z) \lor P(z,a)$. Again, the two literals are incomparable, hence maximal, generating, e.g., $P(a,z') \lor P(z',z) \lor P(z,a)$. This process can be continued forever.

Example 3.15.6. Termination of Standard Tableau on BS Consider the clause set

$$N = \{\neg P(x,y) \lor P(x,z) \lor P(z,y),\ P(a,a)\}$$

After one α-Expansion leading to the branch $(N, \neg P(x,y) \lor P(x,z) \lor P(z,y), P(a,a))$ there is only one γ-Expansion possible leading to $(N, \neg P(x,y) \lor P(x,z) \lor P(z,y), P(a,a), \neg P(a,a) \lor P(a,a) \lor P(a,a))$. Now a β-Expansion results eventually in three branches where the branch containing $\neg P(a,a)$ can be closed and the two branches containing $P(a,a)$ remain open. The resulting tableau is saturated.

There is, in theory and in practice, a difference in techniques that are successful in showing termination of calculi on first-order fragments depending on whether $\operatorname{grd}(\Sigma, N)$ is finite or not. For the following fragment, monadic shallow linear Horn clauses (MSLH), $\operatorname{grd}(\Sigma, N)$ is infinite, in general. Superposition terminates on this fragment, while no tableau calculus terminates.

Definition 3.15.7 (Monadic Shallow Linear Horn Clause)**.** A clause $\neg P_1(\vec{t_1}) \lor \ldots \lor \neg P_n(\vec{t_n}) \lor Q_1(\vec{s_1}) \lor \ldots \lor Q_m(\vec{s_m})$ is monadic shallow linear Horn, if

1. $m \leq 1$, i.e., the clause is Horn,
2. all P_i, Q_j are monadic predicates,
3. if $m = 1$ then s_1 is a shallow, linear term $f(x_1, \ldots, x_n)$ or a ground term.

Now a clause set N is belongs to the *MSLH* fragment, if all its clauses are monadic shallow linear horn clauses. Note that there are no restrictions on the structure of terms in negative literals.

Theorem 3.15.8 (Unsatisfiability of MSLH Clause Sets is Decidable)**.** The unsatisfiability of a set of MSLH clauses is decidable.

Proof. By termination of superposition. For the proof I assume a KBO, all symbols weight one, and function symbols are larger in the precedence than predicate symbols. Furthermore, the following selection strategy is needed for termination. In any clause $\neg P(t) \lor C$ the literal $\neg P(t)$ is selected, if

(i) t is a non-variable term,

(ii) or there is no non-variable term in a negative literal, and t is a variable that does not occur in the positive literal of C,

(iii) or all negative literals have variable arguments but C does not contain any positive literal.

The effect of this strategy is that clauses where no negative literal is selected have on of the following two forms:

$$\neg P_1(x_1) \lor \ldots \lor \neg P_n(x_n) \lor Q(f(x_1, \ldots, x_n))$$

where not all x_i of $Q(f(x_1, \ldots, x_n))$ necessarily occur in a negative literal and there may be several negative literals having some x_j as its argument. For this clause $Q(f(x_1, \ldots, x_n))$ is the only strictly maximal literal. The second form is

$$\neg P_1(x) \lor \ldots \lor \neg P_n(x) \lor Q(x)$$

where maximality depends on the precedence on the predicates. Importantly, there are only finitely many different clauses of the above two forms with respect to condensation and subsumption. If the positive Q contains a non-constant ground term initially, this can also be transformed into a finite set of equivalent clauses of the above first form. In addition, only for the above two forms a positive literal can become maximal and therefore be used in a superposition inference. Then any superposition inference generates either a clause of the above form, and there are only finitely many, or the resulting clause is strictly smaller with respect to the multiset of all subterms of the parent clause that has not the above form. □

3.16 Decision Procedures for the Bernays-Schönfinkel (BS) Fragment

In Section 3.15 I showed that unsatisfiability (validity) of first-order logic (clause sets) is undecidable, Theorem 3.15.1. So decision procedures can only exists for fragments, e.g., the Bernays-Schönfinkel (BS) or the MSLH fragment introduced in Section 3.15. This section presents several decision procedures for the BS fragment. Some of them can be extended to complete calculi for full first-order logic and some are refinements of full first-order logic calculi.

Historically, the BS fragment has been defined as all first-order sentences of the form $\exists\vec{x}.\forall\vec{y}.\phi$ where ϕ is quantifier free and does not contain constant nor function symbols, Definition 3.15.3. After Skolemization, satisfiability is equivalent to the formula $\forall^*\vec{y}.(\phi\{x_1 \mapsto a_1, \ldots, x_n \mapsto a_n\})$ for (fresh) constants $a_1, \ldots, a_n$ which can then be further transformed into CNF.

Thus the Herbrand domain of a BS clause set N is finite, consisting of all constants in N. So is the equisatisfiable set $\operatorname{grd}(N)$ where satisfiability can then be decided by any decision procedure for propositional logic. However, the set

$\operatorname{grd}(N)$ is exponentially larger than N, in general. If k is the maximal number of variables of a clause in N, and n the number of different constants in N, then worst-case $|\operatorname{grd}(N)| = \mathrm{O}(|N| \cdot n^k)$. This motivates research for more flexible calculi without a worst-case initial blow-up.

3.16.1 Superposition

The superposition calculus, Section 3.13, is not a decision procedure for the BS fragment, i.e., it does not necessarily terminate on a clause set N of BS clauses, see also Example 3.15.5. Consider a BS clause set consisting of the following two clauses

$$\begin{array}{ll} 1 & \neg R(x,y) \lor \neg R(y,z) \lor R(x,z) \\ 2 & R(x,y) \lor R(y,x) \end{array}$$

describing a transitive and total relation R. With respect to any ordering stable under substitution, the R literals are all incomparable in their respective clauses. The only way to restrict inferences via the superposition calculus is to select one of the negative literals in the transitivity clause, clause 1. The superposition calculus generates an infinite number of clauses including

$$\begin{array}{ll} N_0 = \{1 : \neg R(x,y) \lor \neg R(y,z) \lor R(x,z),\ 2 : R(x,y) \lor R(y,x)\} & \\ \Rightarrow^{1.1,2.1}_{\mathrm{SUP}} & N_0 \cup \{3 : \neg R(y,z) \lor R(x,z) \lor R(y,x)\} \\ \Rightarrow^{3.1,2.1}_{\mathrm{SUP}} & N_1 \cup \{4 : R(x,z) \lor R(y,x) \lor R(z,y)\} \\ \Rightarrow^{4.1,4.2}_{\mathrm{SUP}} & N_2 \cup \{5 : R(x,x)\} \\ \Rightarrow^{4.1,3.1}_{\mathrm{SUP}} & N_3 \cup \{6 : R(x,z) \lor R(y,x) \lor R(y',y) \lor R(z,y')\} \\ \vdots & \end{array}$$

The crucial point is that neither clause 4 nor clause 6 is redundant because of the underlying variable chains. The variable chain can be extended generating clauses of length five, six, ... Obviously, such a clause containing a variable chain contributes a refutation at most of the length of square of the number of different constant symbols. Otherwise, it becomes redundant by the existence of shorter clauses. So one way to turn superposition into a decision procedure for the BS class is to add an additional condensation rule that unifies literals in clauses as soon as all potential ground instantiations with constants yield duplicates.

Condensation-BS $(N \uplus \{L_1 \lor \cdots \lor L_n\}) \Rightarrow_{\mathrm{SUP}} (N \cup \{\operatorname{rdup}((L_1 \lor \ldots \lor L_n)\sigma_{i,j}) \mid \text{if } L_i,\, L_j \text{ are unifiable and } \sigma_{i,j} = \operatorname{mgu}(L_i, L_j)\})$

provided any ground instance $(L_1 \lor \cdots \lor L_n)\delta$ contains at least two duplicate literals

Another way to prevent non-termination is by preventing the generation of arbitrary long clauses. This can be done by a special splitting rule, that splits non-Horn clauses into their Horn parts through instantiation. Assuming two constants a, b for the above example, then clause 2 is replaced by clauses

$$\begin{array}{ll} 2.1 & R(a,b) \lor R(b,a) \\ 2.2 & R(a,a) \\ 2.3 & R(b,b). \end{array}$$

Next the clause $R(a,b) \lor R(b,a)$ can be split, similar to a β-rule application of tableau, resulting in two clause sets

$$\begin{array}{ll} M_1 & = \{\neg R(x,y) \lor \neg R(y,z) \lor R(x,z),\ R(a,a),\ R(b,b),\ R(a,b)\} \\ M_2 & = \{\neg R(x,y) \lor \neg R(y,z) \lor R(x,z),\ R(a,a),\ R(b,b),\ R(b,a)\}. \end{array}$$

Now the original clause set N_0 is satisfiable iff M_1 or M_2 are satisfiable. Employing a rigorous selection strategy where in every clause containing negative literals one negative literal is selected, the superposition calculus will always infer from a positive unit clause and a clause containing at least one negative literal a shorter clause. So it will terminate.

A state is now a set of clause sets. Let k be the number of different constants $a_1, \ldots, a_k$ in the initial clause set N. Then the initial state is the set $M = \{N\}$, Superposition Left is adopted to the new setting, Factoring is no longer needed and the rules Instantiate and Split are added. The variables $x_1, \ldots, x_k$ constitute a *variable chain* between literals L_1, L_k inside a clause C, if there are literals $\{L_1, \ldots, L_k\} \subseteq C$ such that $x_i \in (\text{vars}(L_i) \cap \text{vars}(L_{i+1}))$, $1 \leq i < k$.

Superposition-BS $M \uplus \{N \uplus \{P(t_1, \ldots, t_n), C \lor \neg P(s_1, \ldots, s_n)\}\} \Rightarrow_{\text{SUPBS}}$ $M \cup \{N \cup \{P(t_1, \ldots, t_n), C \lor \neg P(s_1, \ldots, s_n)\} \cup \{C\sigma\}\}$
where (i) $\neg P(s_1, \ldots, s_n)$ is selected in $(C \lor \neg P(s_1, \ldots, s_n))\sigma$ (ii) σ is the mgu of $P(t_1, \ldots, t_n)$ and $P(s_1, \ldots, s_n)$ (iii) $C \lor \neg P(s_1, \ldots, s_n)$ is a Horn clause

Instantiation $M \uplus \{N \uplus \{C \lor A_1 \lor A_2\}\} \Rightarrow_{\text{SUPBS}} M \cup \{N \cup \{(C \lor A_1 \lor A_2)\sigma_i \mid \sigma_i = \{x \mapsto a_i\}, 1 \leq i \leq k\}\}$
where x occurs in a variable chain between A_1 and A_2

Split $M \uplus \{N \uplus \{C_1 \lor A_1 \lor C_2 \lor A_2\}\} \Rightarrow_{\text{SUPBS}} M \cup \{N \cup \{C_1 \lor A_1\}, N \cup \{C_2 \lor A_2\}\}$
where $\text{vars}(C_1 \lor A_1) \cap \text{vars}(C_2 \lor A_2) = \emptyset$

As usual, the clause parts C, C_1, C_2 may be empty. Note that after exhaustive application of Instantiation and Split, every clause purely containing positive literals is a unit clause. This together with the below rigorous selection strategy justifies the strong side conditions of Superposition BS compared to Superposition Left and explains why Factoring is not needed.

Definition 3.16.1 (Rigorous Selection Strategy)**.** A selection strategy is *rigorous* if in any clause containing a negative literal, a negative literal is selected.

Lemma 3.16.2 (SUPBS Basic Properties)**.** The SUPBS rules have the following properties:

1. Superposition BS is sound.
2. Instantiation is sound and complete.
3. Split is sound and complete.

Proof. 1. Follows from the soundness of Superposition Left.

2. Soundness follows from the soundness of variable instantiation. Completeness follows from the fact that $\text{grd}(C \lor A_1 \lor A_2) = \text{grd}(\{(C \lor A_1 \lor A_2)\sigma_i \mid \sigma_i = \{x \mapsto a_i\}, 1 \leq i \leq k\}\})$.

3. I prove $N \uplus \{C_1 \lor A_1 \lor C_2 \lor A_2\}$ is satisfiable iff $N \cup \{C \lor A_1\}$ or $N \cup \{C_2 \lor A_2\}$ is satisfiable. The direction from right to left is obvious, because both $C_1 \lor A_1$ and $C_2 \lor A_2$ subsume $C_1 \lor A_1 \lor C_2 \lor A_2$. For the other direction assume an interpretation $\mathcal{A}$ satisfying $N \uplus \{C_1 \lor A_1 \lor C_2 \lor A_2\}$, in particular $\mathcal{A} \models C_1 \lor A_1 \lor C_2 \lor A_2$. This means for any valuation β we have $\mathcal{A}, \beta \models C_1 \lor A_1 \lor C_2 \lor A_2$. The sub-clauses $C_1 \lor A_1$, $C_2 \lor A_2$ are variable disjoint so β can be split into β_1 assigning values to variables in $C_1 \lor A_1$ and β_2 assigning values to variables in $C_2 \lor A_2$. Then $\mathcal{A}, \beta \models C_1 \lor A_1 \lor C_2 \lor A_2$ for all β iff $\mathcal{A}, \beta_1\beta_2 \models C_1 \lor A_1 \lor C_2 \lor A_2$ for all β_1, β_2 iff $\mathcal{A}, \beta_1 \models C_1 \lor A_1$ or $\mathcal{A}, \beta_2 \models C_2 \lor A_2$ for all β_1, β_2.

□

3.16.2 Non-Redundant Clause Learning (NRCL)

A pair $C \cdot \sigma$ where σ is grounding is called a *closure*. The semantics of a closure $C \cdot \sigma$ is the ground clause $C\sigma$.

The NRCL calculus is a generalization of the CDCL calculus, Section 2.9, to the BS fragment. The BS fragment can be finitely grounded, but with a worst-case exponential blow up. So an obvious procedure would be to perform the grounding and then run CDCL on the resulting first-order ground clauses. Every first-order ground atom then corresponds to a propositional variable. However, in general, it is not wise to incorporate into an automated reasoning calculus a worst-case exponential preprocessing step. Therefore, NRCL rather lifts the CDCL calculus to the first-order BS fragment.

Similar to a CDCL state, an *NRCL state* is a five tuple $(\Gamma; N; U; j; C)$, where Γ is a (partial) model assumption build from ground literals, N the initial BS clause set, U the set of learned BS clauses, j the current level and C is either $\top$, $\bot$ or a BS clause. Literals $L \in \Gamma$ are either annotated with a number, a level, i.e., they have the form L^k meaning that L is the $k-th$ guessed decision literal, or they are annotated with a pair consisting of a clause and a (ground) substitution $L\sigma^{(C \lor L)\cdot\sigma}$ that forced the literal to become true. A pair $(C \cdot \sigma)$