be lifted to inferences, in the propositional case to Superposition Left applications. A Superposition Left inference

$$(N \uplus \{C_1 \lor P, C_2 \lor \neg P\}) \Rightarrow_{\text{SUP}} (N \cup \{C_1 \lor P, C_2 \lor \neg P\} \cup \{C_1 \lor C_2\})$$

is redundant if either one of the clauses $C_1 \lor P, C_2 \lor \neg P$ is redundant, or if $N^{\prec C_2 \lor \neg P} \models C_1 \lor C_2$. For a Factoring inference, the conclusion $C \lor P$ makes the premise $C \lor P \lor P$, so it is sufficient to require that $C \lor P \lor P$ is not redundant in order to guarantee $C \lor P$ to be non-redundant.

Definition 2.7.8 (Saturation)**.** A set N of clauses is called *saturated up to redundancy*, if any inference from non-redundant clauses in N yields a redundant clause with respect to N or is already contained in N.

Alternatively, saturation can be defined on the basis of redundant inferences. An superposition inference is called *redundant* if the inferred clause is redundant with respect to all clauses smaller than the maximal premise of the inference. Then a set N is saturated up to redundancy if all inferences from clauses from N are redundant.

Examples for specific redundancy rules that can be efficiently decided and are already well-known from the resolution calculus, Section 2.6, are

Subsumption	$(N \uplus \{C_1, C_2\}) \;\Rightarrow_{\text{SUP}}\; (N \cup \{C_1\})$
provided $C_1 \subset C_2$	
Tautology Deletion	$(N \uplus \{C \lor P \lor \neg P\}) \;\Rightarrow_{\text{SUP}}\; (N)$
Condensation	$(N \uplus \{C_1 \lor L \lor L\}) \;\Rightarrow_{\text{SUP}}\; (N \cup \{C_1 \lor L\})$
Subsumption Resolution	$(N \uplus \{C_1 \lor L, C_2 \lor \text{comp}(L)\}) \;\Rightarrow_{\text{SUP}}\; (N \cup \{C_1 \lor L, C_2\})$
where $C_1 \subseteq C_2$	

A clause C where Condensation is not applicable is called *condensed.*

Proposition 2.7.9. All clauses removed by Subsumption, Tautology Deletion, Condensation and Subsumption Resolution are redundant with respect to the kept or added clauses.

Corollary 2.7.10 (Soundness)**.** Superposition is sound.

Superposition is a refinement of resolution, so soundness is a consequence of the soundness part of Theorem 2.6.1.

Theorem 2.7.11 (Completeness)**.** If N is saturated up to redundancy and $\bot \notin N$ then N is satisfiable and $N_{\mathcal{I}} \models N$.

Proof. The proof is by contradiction. So I assume: (i) for any clause D derived by Superposition Left or Factoring from N that D is redundant, i.e., $N^{\prec D} \models D$, (ii) $\bot \notin N$ and (iii) $N_{\mathcal{I}} \not\models N$. Then there is a minimal, with respect to $\prec$, clause

$C \lor L \in N$ such that $N_{\mathcal{I}} \not\models C \lor L$ and L is a selected literal in $C \lor L$ or no literal in $C \lor L$ is selected and L is maximal. This clause must exist because $\bot \notin N$.

The clause $C \lor L$ is not redundant. For otherwise, $N^{\prec C \lor L} \models C \lor L$ and hence $N_{\mathcal{I}} \models C \lor L$, because $N_{\mathcal{I}} \models N^{\prec C \lor L}$, a contradiction.

I distinguish the case L is a positive and no literal selected in $C \lor L$ or L is a negative literal. Firstly, assume L is positive, i.e., $L = P$ for some propositional variable P. Now if P is strictly maximal in $C \lor P$ then actually $\delta_{C \lor P} = \{P\}$ and hence $N_{\mathcal{I}} \models C \lor P$, a contradiction. So P is not strictly maximal. But then actually $C \lor P$ has the form $C_1' \lor P \lor P$ and Factoring derives $C_1' \lor P$ where $(C_1' \lor P) \prec (C_1' \lor P \lor P)$. Now $C_1' \lor P$ is not redundant, strictly smaller than $C \lor L$, we have $C_1' \lor P \in N$ and $N_{\mathcal{I}} \not\models C_1' \lor P$, a contradiction against the choice that $C \lor L$ is minimal.

Secondly, let us assume L is negative, i.e., $L = \neg P$ for some propositional variable P. Then, since $N_{\mathcal{I}} \not\models C \lor \neg P$ we know $P \in N_{\mathcal{I}}$. So there is a clause $D \lor P \in N$ where $\delta_{D \lor P} = \{P\}$ and P is strictly maximal in $D \lor P$ and $(D \lor P) \prec (C \lor \neg P)$. So Superposition Left derives $C \lor D$ where $(C \lor D) \prec (C \lor \neg P)$. The derived clause $C \lor D$ cannot be redundant, because for otherwise either $N^{\prec D \lor P} \models D \lor P$ or $N^{\prec C \lor \neg P} \models C \lor \neg P$. So $C \lor D \in N$ and $N_{\mathcal{I}} \not\models C \lor D$, a contradiction against the choice that $C \lor L$ is the minimal false clause. $\square$

So the proof actually tells us that at any point in time we need only to consider either a superposition left inference between a minimal false clause and a productive clause or a factoring inference on a minimal false clause.

The proof relies on the abstract redundancy notion and not on the specific redundancy rules introduced above. However, it also goes through on the basis of the concrete redundancy notions, see Exercise **??**.

According to Theorem 2.7.11 if a clause set N is saturated up to redundancy, the interpretation $N_{\mathcal{I}}$ is a model for N. This does not hold the other way round. If $N_{\mathcal{I}}$ is a model for N then N is not saturated, in general, see Exercise **??**.

I mentioned already that the abstract redundancy notion of superposition goes beyond the classical resolution reduction rules tautology deletion, subsumption, subsumption resolution and condensation. For example consider the clause set

$$N = \{\neg S \lor P,\ S \lor Q \lor \neg R,\ P \lor Q \lor \neg R\}$$

with ordering $S \prec P \prec Q \prec R$. Then $N^{\prec P \lor Q \lor \neg R} \models P \lor Q \lor \neg R$, i.e., the clause $P \lor Q \lor \neg R$ is redundant and can be deleted. This deletion is not justified by any of the classical resolution reduction rules.

$\boxed{\mathtt{I}}$

In practice, there is a tradeoff between the unsuccessful testing of a powerful redundancy notion and keeping redundant clauses. Already in propositional logic there are exponentially many resolution or superposition inferences possible for a clause set. Testing the abstract superposition redundancy notion requires exponential run time in the size of the clause set $N^{\prec C}$. Inferences generated with respect to the partial model operator $N_{\mathcal{I}}$

following the proof of Theorem 2.7.11 are provably non-redundant with respect to the abstract superposition redundancy notion. Actually, designing a propositional theorem proving algorithm following the proof of Theorem 2.7.11 results in a deterministic system, without any choices once the atom ordering $\prec$ is fixed. Unfortunately, the resulting system is not very powerful in practice, because it cannot adopt to the problem structure. Please recall that the minimal literals in the ordering are then always highly preferred in the resulting learned clauses.

A calculus nicely demonstrating the tradeoff between restricting inferences and a corresponding redundancy notion preserving completeness is *Lock Resolution* [20]. For lock resolution an ordering is given per literal occurrence in a clause by attaching an index to each individual literal. The literal with the maximal index in a clause is then the maximal literal in that clause. Similar to propositional superposition, inferences are restricted to maximal literals.

Lock Resolution $(N \uplus \{C_1 \lor P^i, C_2 \lor \neg P^j\}) \;\Rightarrow_{\text{LOCK}}\; (N \cup \{C_1 \lor P^i, C_2 \lor \neg P^j\} \cup \{C_1 \lor C_2\})$
where (i) i is a maximal index in $C_1 \lor P^i$ and (ii) j is a maximal index in $C_2 \lor \neg P^j$

Lock Factoring $(N \uplus \{C \lor P^i \lor P^j\}) \;\Rightarrow_{\text{LOCK}}\; (N \cup \{C \lor P^i \lor P^j\} \cup \{C \lor P^j\})$
where j is a maximal index in in $C \lor P^i \lor P^j$

The below Example 2.7.12 demonstrates that for lock resolution there is no compatible redundancy notion in the sense that even tautologies must not be removed.

Example 2.7.12 (Lock Resolution)**.** Consider the unsatisfiable clause set

$$\begin{array}{ll} P^1 \lor Q^2 & \neg P^3 \lor \neg Q^4 \\ \neg Q^5 \lor P^6 & Q^7 \lor \neg P^8 \end{array}$$

over propositional variables P, Q. There are only two possible lock resolution inferences between the two clauses in the first row and the two clauses in the second row, respectively. They lead to the two tautologies $P^1 \lor \neg P^3$ and $\neg Q^5 \lor Q^7$.

Still, lock resolution is complete. It is just that all the well-known redundancy criteria compatible with resolution or superposition are not compatible with lock resolution.

2.8 Davis Putnam Logemann Loveland Procedure (DPLL)

A DPLL problem state is a pair $(M; N)$ where M a sequence of partly annotated literals, and N is a set of clauses. In particular, the following states can be distinguished:

MiniSat [38] combines Restart and Forget as follows: there is always a limit c for the number of learned clauses. If c clauses are learned, then they are sorted with respect to an activity score. The $\frac{c}{2}$ clauses with lowest score are thrown away, c is increased by a constant and a Restart is performed. Recall that performing a restart is needed to clear the trail. The VSIDS heuristic together with phase saving directs the search towards the same state that was generated before the restart.

2.10.5 The Overall Algorithm and Further Heuristics & Strategies

Algorithm 5 presents a CDCL solver including most aspects discussed in previous sections. It implements a reasonable strategy and includes the incorporation of the VSIDS heuristic and restarts. It does not contain a heuristic for an initial VSIDS score. Typical solutions are to start with a score of 0 for all variables or to start with the number of variable occurrences in N. Similarly for an application of the rule Decide. For a variable with maximal VSIDS score either the positive or the negative literal can be decided. Again this can be implemented via a heuristic on the number of literal occurrences in N. Important is *phase saving*: once a literal has been decided, after removal from the trail due to Restart or Backtrack, if it is decided again, it is decided with the same sign.

The restart heuristic typically considers also unit clauses. Once a unit clause is learned a restart is performed immediately. Unit clauses always propagate, so their literals are collected during a run at the start of the trail. This applies as well to literals propagating solely because of unit clauses, i.e., at level 0.

2.11 Superposition and CDCL

At the time of this writing it is often believed that the superposition (resolution) calculus is not a good choice on SAT problems in practice. Most of the successful SAT solvers implemented in 2015 are based on CDCL. In this section I will develop some relationships between superposition and CDCL. Actually, CDCL can be considered as a superposition calculus with respect to a generalized model operator.

The goal of the original model operator, Definition 2.7.6, is to create minimal models with respect to positive literals, i.e., if $N_{\mathcal{I}} \models N$ for some N, then there is no $M' \subset N_{\mathcal{I}}$ such that $M' \models N$. However, if the goal generating minimal models is dropped, then there is more freedom to construct models while preserving the general properties of the superposition calculus, in particular, the notion of redundancy. The gained freedom can be used to be more flexible when generating a partial model with respect to a given set of clauses. For example, consider the two clauses in the clause set

$$N = \{P \vee Q,\ \neg P \vee R\}$$

Algorithm 5: CDCL(S)

```
Input  : An initial state $(\epsilon; N; \emptyset; 0; \top)$.
Output: A final state $S = (M; N; U; k; \top)$ or $S = (M; N; U; k; \bot)$
while (any rule applicable) do
    ifrule (Conflict($S$)) then
        while (Skip($S$) || Resolve($S$)) do
            update VSIDS scores on resolved literals;
        update VSIDS scores on learned clause;
        Backtrack($S$);
        if (potential VSIDS score overflow) then
            scale VSIDS scores;
        if (forget heuristic) then
            Forget($S$) clauses ;
            Restart($S$);
        else
            if (restart heuristic) then
                Restart($S$);
    else
        ifrule (!Propagate($S$)) then
            Decide($S$) literal with max. VSIDS score;
return($S$);
```

with precedence $R \prec Q \prec P$. The superposition model operator generates $N_{\mathcal{I}} = \{P\}$ which is not a model for N. However, this model can be extended to a model for N by adding R to it. The superposition model operator does not include R because it is not maximal in the second clause. Starting with a decision on P, the CDCL calculus derives the model P, R via propagation. In the sequel, I show that a generalized superposition model operator can in fact generate this model as well.

In addition to an ordering $\prec$ I assume a decision heuristic $\mathcal{H}$ that selects whether a literal should be productive, i.e., included in the model, or not.

Definition 2.11.1 (Heuristic-Based Partial Model Construction)**.** Given a clause set N, a set of propositional variables $M \subseteq \Sigma$, a total ordering $\prec$, and a variable heuristic $\mathcal{H} : \Sigma \to \{0, 1\}$, the (partial) model $N_M^{\mathcal{H}}$ for N with $P, Q \in M$

is inductively constructed as follows:

$$
\begin{array}{lcl}
N_P^{\mathcal{H}} & := & \bigcup_{Q \prec P} \delta_Q^{\mathcal{H}} \\[1em]
\delta_P^{\mathcal{H}} & := & \begin{cases} \{P\} & \text{if there is a clause } (D \lor P) \in N, \text{ such that } N_P^{\mathcal{H}} \models \neg D \\ & \text{and either } P \text{ is strictly maximal} \qquad \text{or} \\ & \mathcal{H}(P) = 1 \text{ and there is no clause } (D' \lor \neg P) \in N, D' \prec P \\ & \text{such that } N_P^{\mathcal{H}} \models \neg D' \\ \emptyset & \text{otherwise} \end{cases} \\[1em]
N_M^{\mathcal{H}} & := & \bigcup_{P \in M} \delta_P^{\mathcal{H}}
\end{array}
$$

In case a clause $(D \lor P) \in N$ is the reason for $\delta_P^{\mathcal{H}} = \{P\}$ acoording to the above Definition 2.11.1, we say that $(D \lor P)$ *produced* P in $\delta_P^{\mathcal{H}}$, $N_M^{\mathcal{H}}$, and the clause $(D \lor P)$ is *productive*, analogous to Definition 2.7.6.

Please note that $N_{\mathcal{I}}$ is defined inductively over the clause ordering $\prec$ whereas $N_M^{\mathcal{H}}$ is defined inductively over the atom ordering $\prec$. For each atom P, the heuristic model construction $N_M^{\mathcal{H}}$ considers all clauses with maximal P, $\neg P$ at once. The set M of propositional variables may and will often equate Σ.

The heuristic-based model operator $N_M^{\mathcal{H}}$ enjoys many properties of the standard model operator $N_{\mathcal{I}}$ and generalizes it.

Lemma 2.11.2. If $\mathcal{H}(P) = 0$ for all $P \in \Sigma$ then $N_{\mathcal{I}} = N_\Sigma^{\mathcal{H}}$ for any N.

Proof. The proof is by contradiction. Assume $N_{\mathcal{I}} \neq N_\Sigma^{\mathcal{H}}$, i.e., there is a minimal $P \in \Sigma$ such that P occurs only in one set out of $N_{\mathcal{I}}$ and $N_\Sigma^{\mathcal{H}}$.

Case 1: $P \in N_{\mathcal{I}}$ but $P \notin N_\Sigma^{\mathcal{H}}$.
Then there is a productive clause $D = D' \lor P \in N$ such that P is strictly maximal in this clause and $N_D \models \neg D'$. Since P is strictly maximal in D the clause D' only contains literals strictly smaller than P. Since both interpretations agree on all literals smaller than P from $N_D \models \neg D'$ it follows $N_P^{\mathcal{H}} \models \neg D'$ and therefore $\delta_P^{\mathcal{H}} = \{P\}$ contradicting $P \notin N_\Sigma^{\mathcal{H}}$.

Case 2: $P \notin N_{\mathcal{I}}$ but $P \in N_\Sigma^{\mathcal{H}}$.
Then there is a minimal productive clause $D = D' \lor P \in N$ such that P is strictly maximal in this clause and $N_P^{\mathcal{H}} \models \neg D'$ because $\mathcal{H}(P) = 0$. The atom P is strictly maximal in D, so the clause D' only contains literals strictly smaller than P. Since both interpretations agree on all literals smaller than P from $N_P^{\mathcal{H}} \models \neg D'$ it follows $N_D \models \neg D'$ and therefore $\delta_D = \{P\}$ contradicting $P \notin N_{\mathcal{I}}$. $\square$

So the new model operator $N_M^{\mathcal{H}}$ is a generalization of $N_{\mathcal{I}}$. Next, I will show that with the help of $N_M^{\mathcal{H}}$ a close relationship between the model assumptions generated by the CDCL calculus and the superposition model operator can be established. This result can then further be used to apply the abstract superposition redundancy criteria to CDCL. But before going into the relationship I first show that the generalized superposition partial model operator $N_\Sigma^\mathcal{H}$ supports the standard superposition completeness result, analogous to Theorem 2.7.11. Recall that the same notion of redundancy, Definition 2.7.4, is used.

Theorem 2.11.3. If N is saturated up to redundancy and $\bot \notin N$ then N is satisfiable and $N_\Sigma^\mathcal{H} \models N$.

Proof. The proof is by contradiction. So I assume: (i) any clause C derived by Superposition Left or Factoring from N is redundant, i.e., $N^{\prec C} \models C$, (ii) $\bot \notin N$ and (iii) $N_\Sigma^\mathcal{H} \not\models N$. Then there is a minimal, with respect to $\prec$, clause $C_1 \lor L \in N$ such that $N_\Sigma^\mathcal{H} \not\models C_1 \lor L$ and L is a maximal literal in $C_1 \lor L$. This clause must exist because $\bot \notin N$.

The clause $C_1 \lor L$ is not redundant. For otherwise, $N^\mathcal{H}_{\text{atom}(L)} \cup \delta_P^\mathcal{H} \models C_1 \lor L$ and hence $N_\Sigma^\mathcal{H} \models C_1 \lor L$, a contradiction.

I distinguish the case whether L is a positive or a negative literal. Firstly, assume L is positive, i.e., $L = P$ for some propositional variable P. Now if P is strictly maximal in $C_1 \lor P$ then actually $\delta_P^\mathcal{H} = \{P\}$ and hence $N_\Sigma^\mathcal{H} \models C_1 \lor P$, a contradiction. So P is not strictly maximal. But then actually $C_1 \lor P$ has the form $C_1' \lor P \lor P$ and Factoring derives $C_1' \lor P$ where $(C_1' \lor P) \prec (C_1' \lor P \lor P)$. The clause $C_1' \lor P$ is not redundant, strictly smaller than $C_1 \lor L$, we have $C_1' \lor P \in N$ and $N_\Sigma^\mathcal{H} \not\models C_1' \lor P$, a contradiction against the choice that $C_1 \lor L$ is minimal.

Secondly, assume L is negative, i.e., $L = \neg P$ for some propositional variable P. Then, since $N_\Sigma^\mathcal{H} \not\models C_1 \lor \neg P$ we know $P \in N_\Sigma^\mathcal{H}$, i.e., $\delta_P^\mathcal{H} = \{P\}$. There are two cases to distinguish. Firstly, there is a clause $C_2 \lor P \in N$ where P is strictly maximal, $N_P^\mathcal{H} \models \neg C_2$, and by definition $(C_2 \lor P) \prec (C_1 \lor \neg P)$. Since $C_1 \prec \neg P$ and $C_1 \lor \neg P$ is not a tautology, it holds $C_1 \prec P$. So a Superposition Left inference derives $C_1 \lor C_2$ where $(C_1 \lor C_2) \prec (C_1 \lor \neg P)$. The derived clause $C_1 \lor C_2$ cannot be redundant, because for otherwise either $N_P^\mathcal{H} \models C_2$ or $N_P^\mathcal{H} \models C_1$. So $C_1 \lor C_2 \in N$ and $N_\Sigma^\mathcal{H} \not\models C_1 \lor C_2$, a contradiction against the choice that $C_1 \lor L$ is minimal. Secondly, there is no clause $C_2 \lor P \in N$ where P is strictly maximal but $\mathcal{H}(P) = 1$. But a further condition for this case is that there is no clause $(C_1 \lor \neg P) \in N$, $\neg P$ strictly maximal in $C_1 \lor \neg P$, because the clause is condensed, such that $N_P^\mathcal{H} \not\models C_1$ contradicting the above choice of $C_1 \lor \neg P$. ∎

Recalling Section 2.7, Superposition is based on an ordering $\prec$. It relies on a model assumption $N_\mathcal{I}$, Definition 2.7.6, or its generalization $N_\Sigma^\mathcal{H}$, Definition 2.11.1. Given a set N of clauses, either $N_\mathcal{I}$ ($N_\Sigma^\mathcal{H}$) is a model for N, N contains the empty clause, or there is a superposition inference on the minimal false clause with respect to $\prec$, see the proof of Theorem 2.7.11 or Theorem 2.11.3, respectively.

CDCL is based on a variable selection heuristic. It computes a model assumption via decision variables and propagation. Either this assumption is a model of N, N contains the empty clause, or there is a backjump clause that is learned.

For a CDCL state (M, N, U, k, D) generated by an application of the rule Conflict, where $M = L_1, \ldots, L_n$ any following Resolve step actually corresponds to a superposition step between a minimal false clause and its productive counterpart, with respect to the precedence $\text{atom}(L_1) \prec \text{atom}(L_2) \prec \ldots \prec \text{atom}(L_n)$. Furthermore, the decision heuristic $\mathcal{H}$ is defined by $\mathcal{H}(\text{atom}(L_m)) = 1$ if there is a positive decision literal L_m^k occurring in M and $\mathcal{H}(\text{atom}(L_m)) = 0$ otherwise. Then the learned CDCL clause is in fact generated by a superposition inference with respect to the model operator $N_\Sigma^{\mathcal{H}}$. The following propositions present this relationship between Superposition and CDCL in full detail.

Theorem 2.11.4. Let $(M, N, U, k, C \lor K)$ be a CDCL state generated by rule Conflict and a reasonable strategy where $M = L_1, \ldots, L_n$. Let $\mathcal{H}(\text{atom}(L_m)) = 1$ for any positive decision literal L_m^i occurring in M and $\mathcal{H}(\text{atom}(L_m)) = 0$ otherwise. Furthermore, I assume that if CDCL can propagate both P and $\neg P$ in some state, then it propagates P. The superposition precedence is $\text{atom}(L_1) \prec \text{atom}(L_2) \prec \ldots \prec \text{atom}(L_n)$. Let K be maximal in $C \lor K$ and $C \lor K$ be the minimal false clause with respect to $\prec$. Then

1. L_n is a propagated literal and $K = \text{comp}(L_n)$.
2. The clause generated by $C \lor K$ and the clause propagating L_n is the result of a Superposition Left inference between the clauses and it is not redundant.
3. $N^{\mathcal{H}}_{\{L_1,\ldots,L_n\}} = \{P \mid P \in M\}$

Proof. 1. Assume $K \neq \text{comp}(L_n)$. Since $C \lor K$ was derived by rule Conflict it is false with respect to M. Since K is maximal in $C \lor K$ it is the complement of some L_i from M with $i < n$ contradicting a reasonable strategy. So $K = \text{comp}(L_n)$. Assume L_n is a decision literal. But then at trail $L_1, \ldots, L_{n-1}$ the clause $C \lor K$ propagates K with respect to $L_1 \ldots L_{n-1}$, so with a reasonable strategy, the literal L_n cannot be a decision literal but its complement was propagated by the clause $C \lor K$.

2. Let $D \lor L_n$ be the clause propagating L_n. Both C and D only contain literals with variables from $\text{atom}(L_1), \ldots, \text{atom}(L_{n-1})$. Since in CDCL duplicate literals are (silently) removed, the literal L_n is strictly maximal in $D \lor L_n$ and $K = \text{comp}(L_n)$ is strictly maximal in $C \lor K$. So resolving on L_n is a superposition inference with respect to the atom ordering $\text{atom}(L_1) \prec \text{atom}(L_2) \ldots \prec \text{atom}(L_n)$. Now assume $C \lor D$ is redundant, i.e., there are clauses $D_1, \ldots, D_n$ from $N \cup U$ with $D_i \prec C \lor D$ and $D_1, \ldots, D_n \models C \lor D$. Since C and D are false in $L_1 \ldots L_{n-1}$, the resolvent $C \lor D$ is false in $L_1 \ldots L_{n-1}$ as well and there is at least one D_i that is also false in $L_1 \ldots L_{n-1}$. A contradiction against the assumption that $L_1 \ldots L_{n-1}$ does not falsify any clause in $N \cup U$, i.e., rule Conflict was not applied eagerly contradicting a reasonable strategy.

3. Firstly, note that if CDCL can propagate both P and $\neg P$ then either way the next applicable reasonable rule is Conflict, so propagating P in favor of $\neg P$

is not a restriction on the propagation order. I prove the equality of the atom sets by induction on n.
"$\supseteq$" For $n = 1$ and $L_1 = [\neg]P$ propagated in M, there are two cases: (i) $L_1 = P$, so $P \in N$ and $\delta_P^{\mathcal{H}} = \{P\}$; (ii) $L_1 = \neg P$, so $\neg P \in N$ and $P \notin N$, therefore $\delta_P^{\mathcal{H}} = \emptyset$. If $L_1 = [\neg]P$ is a decision literal then $\neg P \notin N$ and $P \notin N$. Again there are two cases: (i) $L_1 = P$, so $\mathcal{H}(P) = 1$ and hence $\delta_P^{\mathcal{H}} = \{P\}$; (ii) $L_1 = \neg P$, so $\mathcal{H}(P) = 0$ and hence $\delta_P^{\mathcal{H}} = \emptyset$.

For the step $(n-1) \to n$, I do the same case analysis as for the base case $n = 1$. If $L_n = [\neg]P$ is propagated in M, there are two cases: (i) $L_n = P$, so $D \vee P \in N$ and $L_1 \ldots L_{n-1} \models \neg D$. By induction hypothesis $N^{\mathcal{H}}_{\{L_1,\ldots,L_{n-1}\}} \models \neg D$, $L_i \prec P$, so $\delta_P^{\mathcal{H}} = \{P\}$; (ii) $L_n = \neg P$, so $D \vee \neg P \in N$, $\mathcal{H}(P) = 0$ and there is no clause $D' \vee P \in N$ propagating P, hence $\delta_P^{\mathcal{H}} = \emptyset$. If $L_n = [\neg]P$ is a decision literal in M then due to the reasonable strategy, there is no clause propagating L_n on the basis of the trail $L_1 \ldots L_{n-1}$. Again two cases: (i) $L_n = P$, so $\mathcal{H}(P) = 1$ and there is no clause $C_1 \vee \neg P$ such that $L_1, \ldots, L_{n-1} \models \neg C_1$, hence $\delta_P^{\mathcal{H}} = \{P\}$; (ii) if $L_n = \neg P$, $\mathcal{H}(P) = 0$, and there is no clause $C_1 \vee P$ such that $L_1 \ldots L_{n-1} \models \neg C_1$, so $\delta_P^{\mathcal{H}} = \emptyset$.
"$\subseteq$" By construction. □ □

Example 2.9.9 shows that requiring $C \vee K$ to be minimal, is necessary for its non-redundancy. However, even if $C \vee K$ is chosen non-minimal and hence potentially redundant, the resolvent $C \vee D$ is always non-redundant because of the reasonable strategy. Therefore, choosing a minimal clause potentially reduces the number of Resolve steps but starting with a non-minimal clause will results in eventually learning a non-redundant clause.

Proposition 2.11.5 (Resolve and Skip)**.** Clauses generated by rule Resolve are superposition inferences in the sense of Proposition 2.11.4 and always false with respect to $N_\Sigma^{\mathcal{H}}$.

Proof. for the reasonable CDCL run we assume that if both P and $\neg P$ can be propagated in some state, then P is propagated (this can only be the final propagation before Conflict) □

Proposition 2.11.4 is actually a nice explanation for the efficiency of the CDCL procedure: a learned clause is never redundant. Recall that redundancy here means that the learned clause C is not entailed by smaller clauses in $N \cup U$. Furthermore, the ordering underlying Proposition 2.11.4 is based on the trail, i.e., it changes during a CDCL run. For superposition it is well known that changing the ordering is not compatible with the notion of redundancy, i.e., superposition is incomplete when the ordering may be changed infinitely often and the superposition redundancy notion is applied.

Example 2.11.6. Consider the superposition left inference between the clauses $P \vee Q$ and $R \vee \neg Q$ with ordering $P \prec R \prec Q$ resulting in $P \vee R$. Changing the ordering to $Q \prec P \prec R$ the inference $P \vee R$ becomes redundant. So flipping infinitely often between $P \prec R \prec Q$ and $Q \prec P \prec R$ is already sufficient to prevent any saturation progress.

Although Example 2.11.6 shows that changing the ordering is not compatible with redundancy and superposition completeness, Proposition 2.11.4 proves that any CDCL learned clause is not redundant in the superposition sense and the CDCL procedure changes the ordering and is complete. This relationship shows the power of reasoning with respect to a (partial) model assumption. The model assumption actually prevents the generation of redundant clauses. Nevertheless, also in the CDCL framework completeness would be lost if redundant clauses are eagerly removed in general. Either the ordering is not changed and the superposition redundancy notion can be eagerly applied or only a weaker notion of redundancy is possible while keeping completeness.

The crucial point is that for the superposition calculus the ordering is also the bases for termination and completeness. If the completeness proof can be decoupled from the ordering, then the ordering might be changed infinitely often and other notions of redundancy become available. However, these new notions of redundancy need to be compatible with the completeness and termination proof.

2.12 Implementing Superposition

CDCL can be interpreted as a superposition calculus where the underlying ordering is changed, Theorem 2.11.4. On the other hand, an implementation of the superposition calculus where the ordering is dynamically changed, is no longer complete in the sense of Theorem 2.7.11, see Example 2.11.6.

An implementation of superposition with a fixed ordering but in the style of CDCL model development does not work either. Although the superposition partial model $N^{\mathcal{H}}_{\{L_1,\ldots,L_n\}}$, Theorem 2.11.4, constructed with respect to the atom ordering $\text{atom}(L_1) \prec \text{atom}(L_2) \prec \ldots \prec \text{atom}(L_n)$ coincides with the trail $(M, N, U, k, C \lor K)$, after learning the clause generated out of the conflict clause $C \lor K$ this is no longer the case, in general. A CDCL run will then typically generate a different trail, hence a different superposition ordering will correspond to this trail. Still, superposition can be implemented on the basis of the partial model operator, Algorithm 6.

For the result $C \lor D$ of the Superposition Left inference between $C \lor \neg P$ and $D \lor P$ in Algorithm 6 I assume duplicate literals to be silently removed. This is justified by the reduction rule Condensation. Actually, in an implementation of propositional superposition the reduction rule Condensation replaces Factoring. The clause $C \lor D$ is not redundant but potentially simplifies or reduces clauses in N. For simplification any instance of the abstract redundancy criterion, Definition 2.7.4, may be used. I will further discuss instances of the abstract redundancy notion in Section 2.13. The model operator $N_{\mathcal{I}}$ can be replaced by the model operator $N^{\mathcal{H}}_{\Sigma}$ in Algorithm 6.