where $M = \bot$ if N_α is unsatisfiable and $M = \{L_1, \ldots, L_n\}$ if $\{L_1, \ldots, L_n\}$ is a model for N_α

Instantiate $(N \uplus \{C \vee A, D \vee \neg B\}, M) \Rightarrow_{\text{IGEN}} (N \uplus \{C \vee A, D \vee \neg B, (C \vee A)\sigma, (D \vee \neg B)\sigma\}, \top)$

where $M = \{L_1, \ldots, L_n\}$, $\sigma = \text{mgu}(A, B)$, and σ is a proper instantiator of A or B

It is important that the grounding of N_α is obtained by substituting the same constant α for all variables, for otherwise the calculus becomes incomplete. For example, the two unit clauses $P(x, y); \neg P(x, x)$ are unsatisfiable. A grounding $P(a, b); \neg P(a, a)$ results in the model $\mathcal{I}_{N_\alpha} = \{P(a, b); \neg P(a, a)\}$ but Instantiate is not applicable, because the unifier $\{x \mapsto y\}$ is not a proper instantiater for both literals.

The model M is actually not used in rule Instantiate. The proof of the theorem below, however, shows that it is sufficient to consider a minimal false clause $C \vee A$ or $D \vee \neg B$ with respect to $\mathcal{I}_N$, for the inference.

Theorem 3.16.4 (Completeness of InstGen)**.** Let $(N, \top) \Rightarrow^*_{\text{IGEN}} (N', M)$ and let (N', M) be a final state. If N is satisfiable then $M \neq \bot$ and $\mathcal{I}_{N'} \models N'$.

Proof. Suppose $\mathcal{I}_{N'}$ is not a model for N'. Then there exists a minimal ground closure $C \cdot \gamma$ such that $\mathcal{I}_{N'} \not\models C\gamma$. Obviously, $C \cdot \gamma$ was not productive in $\mathcal{I}_{N'}$. So there is no literal $L \in C$ such that $L\gamma$ is undefined in $\mathcal{I}_{C \cdot \gamma}$ and $L_\alpha \in \mathcal{I}_{N_\alpha}$. However, $K_\alpha \in \mathcal{I}_{N_\alpha}$ for some $K \in C$ because $\mathcal{I}_{N_\alpha} \models N_\alpha$. So $C = C' \vee K$ and $\text{comp}(K\gamma) \in \mathcal{I}_{C \cdot \gamma}$. Therefore, there is some ground instance $D\sigma = (D' \vee K')\sigma$ that produces $\text{comp}(K\gamma)$ in $\mathcal{I}_{C \cdot \gamma}$, i.e., $\text{comp}(K')\sigma = K\gamma$. Let $\rho = \text{mgu}(K, \text{comp}(K'))$ and γ' be a substitution such that $(D' \vee K')\rho\gamma' = D\sigma$ and $(C' \vee K)\rho\gamma' = C\gamma$.

Firstly, ρ is a proper instatiator for K' or K. Assume not, then $K_\alpha = \text{comp}(K'_\alpha)$ so both $K'_\alpha \in \mathcal{I}_{N_\alpha}$ and $\text{comp}(K'_\alpha) \in \mathcal{I}_{N_\alpha}$, a contradiction.

Therefore, secondly, ρ is a proper instatiator for K' or K leading to two cases. If ρ is a proper instatiator for K', i.e., it instantiates a variable in K' with some constant then $(D' \vee K')\rho \cdot \gamma' \prec (D' \vee K') \cdot \sigma$ contradicting that N' is saturated or $(D' \vee K') \cdot \sigma$ is a minimal representation.

If ρ is a proper instatiator for K, it instantiates a variable in K with some constant then $(C' \vee K)\rho \cdot \gamma' \prec (C' \vee K) \cdot \gamma$ contradicting that that N' is saturated or $C \cdot \gamma$ was a minimal ground closure. □

Redundancy can be defined analogously to superposition as well. A ground closure $C \cdot \sigma$ is *redundant* in a clause set N, if there are closures $C_1 \cdot \sigma_1, \ldots, C_n \cdot \sigma_n$ from clauses $C_1, \ldots, C_n$ from N such that $C_i \cdot \sigma_i \prec C \cdot \sigma$ for all i and $C_1\sigma_1, \ldots, C_n\sigma_n \models C\sigma$. A clause C from N is *redundant* if all its ground closures $C \cdot \sigma$ are redundant.

3.16.4 SCL Clause Learning from Simple Models

The basic idea of SCL is to lift the principles of CDCL, Section 2.9, to first-order logic: (i) operating wih respect to a partial model assumption represented by a trail, (ii) learning only non-redundant clauses out of false clauses with respect to the trail, (iii) finding models in case no conflict occurs. It is called clause learning from simple models, because the trail is restricted to ground literals. The motivation fot this it twfold: (i) deciding falsity of a first-order clause with variables can be done practically efficiently and (ii) different ground literals don't have common instances resulting in efficient trail operations. Nevertheless, non-redundant clauses with variables can be learned, by using the grounding falsifying a clause to guide resolution stpes on the level of clauses with variables.

Another issue that needs to be addressed is that first-order Herbrand models are infinite, in general. Here the idea of SCL is to restrict the reasoning with respect to some ground literal β, requiring that any trail literal is smaller to β with respect to some well-founded, total, strict ordering $\prec_\beta$ on ground atoms such that for any ground atom A there are only finitely many ground atoms B with $B \prec_\beta A$ [**?**]. For example, a KBO, Definition 3.11.9, could be used to this end.

Then if some model with respect to β and $\prec_\beta$ is found SCL stops in a *stuck* state. Now this partial, finite model might be contained in some model that eventually satisfies the overall clause set. A question that is undecidable, in general, and is not addressed here although it leads to interesting research questions. Or the restriction to β and $\prec_\beta$ was too restrictive to derive a contradiction. Here a new rule Grow is added that strictly extends β. As first-order unsatisfiability implies unsatisfiability with respect to a finite set of ground instances, a suitably chosen β guarantees the derivation of a contradiction via SCL.

The inference rules of SCL operate on a problem state, a six-tuple $(\Gamma; N; U; \beta; k; D)$ where Γ is a sequence of annotated ground literals, the *trail*; N and U are the sets of *initial* and *learned* clauses; β is a ground literal limiting the size of the trail; k counts the number of decisions; and D is a status closure. A *closure* is denoted as $C \cdot \sigma$ and is a pair of a clause C and a grounding substitution σ. Then D is either true $\top$, false $\bot$, abbreviations for closures $\top \cdot \sigma$, $\bot \cdot \sigma$, respectively, or $C \cdot \sigma$ for a non-empty clause C. Literals in Γ are either annotated with a number, also called 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 closure that propagated the literal to become true. A ground literal L is of *level* i with respect to a problem state $(\Gamma; N; U; \beta; k; D)$ if L or $\mathrm{comp}(L)$ occurs in Γ and the first decision literal left from L ($\mathrm{comp}(L)$) in Γ, including L, is annotated with i. If there is no such decision literal then its level is zero. A ground clause D is of *level* i with respect to a problem state $(\Gamma; N; U; \beta; k; D)$ if i is the maximal level of a literal in D. the level of the empty clause $\bot$ is 0. Recall D is a non-empty closure or $\top$ or $\bot$. Similarly, a trail Γ is of level i if the maximal literal in Γ is of level i.

A literal L is *undefined* in Γ if neither L nor $\mathrm{comp}(L)$ occur in Γ. We omit annotations to trail literals if they play no role in the respective context. Initially,

the state for a first-order clause set N is $(\epsilon; N; \emptyset; \beta; 0; \top)$.

Example 3.16.5 (Exhaustive Propagation)**.** Consider the clause set

$$N = \{P(a),\ \neg P(x) \vee P(g(x))\}.$$

Then even with respect to a ground trail, there are infinitely many propagations pssible:

$$[P(a)^{P(a)\cdot\{\}}, P(g(a))^{\neg P(x)\vee P(g(x))\cdot\{x\mapsto a\}}, P(g(g(a)))^{\neg P(x)\vee P(g(x))\cdot\{x\mapsto g(a)\}}, \ldots]$$

Now consider a clause set

$$N' = \{P(a),\ \neg P(x) \vee P(g(x)),\ Q \vee \neg R,\ Q \vee R,\ \neg Q \vee R,\ \neg Q \vee \neg R\}$$

then exhaustive propagation will prevent the detection of unsatisfiability of N'. Even if we restrict the trail to ground literals smaller β, there will be exponentially many propagations possible, see Example 3.16.23. Therefore, for first-order logic it is essential to provide calculi without exhaustive propagation.

The rules for conflict search are:

Propagate $(\Gamma; N; U; \beta; k; \top) \Rightarrow_{\mathrm{SCL}} (\Gamma, L\sigma^{(C_0\vee L)\delta\cdot\sigma}; N; U; \beta; k; \top)$
provided $C \vee L \in (N \cup U)$, $C = C_0 \vee C_1$, $C_1\sigma = L\sigma \vee \cdots \vee L\sigma$, $C_0\sigma$ does not contain $L\sigma$, δ is the mgu of the literals in C_1 and L, $(C \vee L)\sigma$ is ground, $(C \vee L)\sigma \prec_\beta \{\beta\}$, $C_0\sigma$ is false under Γ, and $L\sigma$ is undefined in Γ

The rule Propagate applies exhaustive factoring to the propagated literal with respect to the grounding substitution σ and annotates the factored clause to the propagation literal on the trail.

Decide $(\Gamma; N; U; \beta; k; \top) \Rightarrow_{\mathrm{SCL}} (\Gamma, L\sigma^{k+1}; N; U; \beta; k+1; \top)$
provided $L \in C$ for a $C \in (N \cup U)$, $L\sigma$ is a ground literal undefined in Γ, and $L\sigma \prec_\beta \beta$

Conflict $(\Gamma; N; U; \beta; k; \top) \Rightarrow_{\mathrm{SCL}} (\Gamma; N; U; \beta; k; D \cdot \sigma)$
provided $D \in (N \cup U)$, $D\sigma$ false in Γ for a grounding substitution σ

These rules construct a (partial) model via the trail Γ for $N \cup U$ until a conflict, i.e., a false clause with respect to Γ is found or all ground atoms smaller β are defined in M and $M \models \mathrm{grd}(N)^{\prec_\beta}$. The above rules always terminate, because there are only finitely many ground literals K with $K \prec_\beta \beta$. Choosing an appropriate β is sufficient for completeness for certain first-order fragments, e.g., the BS fragment. In particular, for any fragment with the finite model property, a decision procedure can be achieved with SCL for appropriate β. In general, the rule Grow [**?**] increasing β is needed for full first-order completeness. In the special case of a unit clause L, the rule Propagate actually annotates the

literal L with a closure of itself. So the propagated literals on the trail are annotated with the respective propagating clause and the decision literals with the respective level. If a conflict is found, it is resolved by the rules below. Before any Resolve step, we assume that the respective clauses are renamed such that they do not share any variables and that the grounding substitutions of closures are adjusted accordingly.

Skip $(\Gamma, L; N; U; \beta; k; D \cdot \sigma) \Rightarrow_{\text{SCL}} (\Gamma; N; U; \beta; k - i; D \cdot \sigma)$
provided $\text{comp}(L)$ does not occur in $D\sigma$, if L is a decision literal then $i = 1$, otherwise $i = 0$

Factorize $(\Gamma; N; U; \beta; k; (D \vee L \vee L') \cdot \sigma) \Rightarrow_{\text{SCL}} (\Gamma; N; U; \beta; k; (D \vee L)\eta \cdot \sigma)$
provided $L\sigma = L'\sigma$, $\eta = \text{mgu}(L, L')$

Resolve $(\Gamma, L\delta^{(C \vee L) \cdot \delta}; N; U; \beta; k; (D \vee L') \cdot \sigma)$
$\Rightarrow_{\text{SCL}}$ $(\Gamma, L\delta^{(C \vee L) \cdot \delta}; N; U; \beta; k; (D \vee C)\eta \cdot \sigma\delta)$
provided $L\delta = \text{comp}(L'\sigma)$, $\eta = \text{mgu}(L, \text{comp}(L'))$

Backtrack $(\Gamma_0, K, \Gamma_1, \text{comp}(L\sigma)^k; N; U; \beta; k; (D \vee L) \cdot \sigma)$
$\Rightarrow_{\text{SCL}}$ $(\Gamma_0; N; U \cup \{D \vee L\}; \beta; j; \top)$
provided $D\sigma$ is of level $i' < k$, and Γ_0, K is the minimal trail subsequence such that there is a grounding substitution τ with $(D \vee L)\tau$ is false in Γ_0, K but not in Γ_0, and Γ_0 is of level j

Please note the corner case of rule Backtrack where $\tau = \sigma$ and $i' = j$. The clause $D \vee L$ added by the rule Backtrack to U is called a *learned clause*. The empty clause $\bot$ can only be generated by rule Resolve or be already present in N, hence, as usual for CDCL style calculi, the generation of $\bot$ together with the clauses in $N \cup U$ represent a resolution refutation. The calculus offers freedom with respect to factorization. Literals in the conflict clause can, but do not have to be factorized. In particular, the Factorize rule may remove duplicate literals. The rule Resolve does not remove the literal resolved upon from the trail. Actually, Resolve is applied as long as the rightmost propagated trail literal occurs in the conflict clause. This literal is eventually removed by rule Skip from the trail.

Example 3.16.6. consider the clause set presented in [**?**]:

$$N = \{D = Q \vee R(a, y) \vee R(x, b), C = Q \vee S(x, y) \vee P(x) \vee P(y) \vee \neg R(x, y)\}$$

and a problem state:
$([\neg P(a)^1, \neg P(b)^2, \neg S(a, b)^3, \neg Q^4, \neg R(a, b)^{C \cdot \{x \mapsto a, y \mapsto b\}}]; N; \emptyset; \neg R(b, b); 4; \top)$
derived by SCL. We assume $\neg R(b, b)$ to the largest literal among all ground instances of P, S, Q, R literals over the constants a, b. The rule Conflict is applicable and yields the conflict state
$(\neg P(a)^1, \neg P(b)^2, \neg S(a, b)^3, \neg Q^4, R(a, b)^{C \cdot \{x \mapsto a, y \mapsto b\}}; N; \emptyset; \neg R(b, b); 4; D \cdot \{x \mapsto a, y \mapsto b\})$

from which we can either learn the clause

$$C_1 = Q \vee S(x,b) \vee P(x) \vee P(b) \vee S(a,y) \vee P(a) \vee P(y)$$

or the clause

$$C_2 = Q \vee S(a,b) \vee P(a) \vee P(b)$$

depending on whether we first resolve or factorize. Note that C_2 does not subsume C_1. Both clauses are non-redundant. In order to learn C_1 we need to resolve twice with $R(a,b)^{C\cdot\{x\mapsto a, y\mapsto b\}}$.

Example 3.16.7. Consider the clause set

$$N = \left\{ \begin{array}{l} D = Q \vee R(a,y) \vee R(x,b) \\ C = Q \vee S(x,y) \vee P(x) \vee P(y) \vee \neg R(x,y) \end{array} \right\}$$

and a problem state:

$$([\neg P(a)^1, \neg P(b)^2, \neg S(a,b)^3, \neg Q^4, \neg R(a,b)^{C\cdot\{x\mapsto a, y\mapsto b\}}]; N; \emptyset; \neg R(b,b); 4; \top)$$

derived by SCL. We assume $\neg R(b,b)$ to the largest literal among all ground instances of P, S, Q, R literals over the constants a, b. The rule Conflict is applicable and yields the conflict state

$$\begin{array}{r} (\neg P(a)^1, \neg P(b)^2, \neg S(a,b)^3, \neg Q^4, R(a,b)^{C\cdot\{x\mapsto a, y\mapsto b\}}; \\ N; \emptyset; \neg R(b,b); 4; D\cdot\{x\mapsto a, y\mapsto b\}) \end{array}$$

from which we can either learn the clause

$$C_1 = Q \vee S(x,b) \vee P(x) \vee P(b) \vee S(a,y) \vee P(a) \vee P(y)$$

or the clause

$$C_2 = Q \vee S(a,b) \vee P(a) \vee P(b)$$

depending on whether we first resolve or factorize. Note that C_2 does not subsume C_1. Both clauses are non-redundant. In order to learn C_1 we need to resolve twice with $R(a,b)^{C\cdot\{x\mapsto a, y\mapsto b\}}$.

The first property we prove about SCL is soundness. We prove it via the notion of a sound state.

Definition 3.16.8 (Sound States)**.** A state $(\Gamma; N; U; \beta; k; D)$ is *sound* if the following conditions hold:

1. Γ is a consistent sequence of annotated ground literals, i.e. for a ground literal L it cannot be that $L \in \Gamma$ and $\neg L \in \Gamma$
2. for each decomposition $\Gamma = \Gamma_1, L\sigma^{C\vee L\cdot\sigma}, \Gamma_2$ we have that $C\sigma$ is false under Γ_1 and $L\sigma$ is undefined under Γ_1, $N \cup U \models C \vee L$,
3. for each decomposition $\Gamma = \Gamma_1, L^k, \Gamma_2$ we have that L is undefined in Γ_1,
4. $N \models U$,

5. if $D = C \cdot \sigma$ then $C\sigma$ is false under Γ and $N \models C$. In particular, $\mathrm{grd}^{\prec_\beta \beta}(N) \models C\sigma$,

6. for any $L \in \Gamma$ we have $L \prec_\beta \beta$ and there is a $C \in N \cup U$ such that $L \in C$.

To show soundness of SCL, we first show soundness of the initial state. Then, we show that all SCL rule applications preserve soundness, which shows soundness of the overall calculus starting from the initial state.

Lemma 3.16.9 (Soundness of the initial state)**.** The initial state $(\epsilon; N; \emptyset; \beta; 0; \top)$ is sound.

Proof. Criteria 1–3 and 6 are trivially satisfied by $\Gamma = \epsilon$. Furthermore, $N \models \emptyset$, fulfilling criterion 4. Lastly, criterion 5 is trivially fulfilled for $D = \top$. □

Theorem 3.16.10 (Soundness of SCL)**.** All SCL rules preserve soundness, i.e. they map a sound state onto a sound state.

Proof. As the hypothesis, assume that a state $(\Gamma; N; U; \beta; k; D)$ is sound. We show that any application of a rule results again in a sound state.

Decide. Assume Decide is applicable to $(\Gamma; N; U; \beta; k; D)$, yielding a resulting state $(\Gamma, L\sigma^{k+1}; N; U; \beta; k+1; D)$. Then there is a $L \in C$ for $C \in N \cup U$, $L\sigma$ is ground and undefined in Γ, and $L\sigma \prec_\beta \beta$. Also, there can be no active conflict, i.e. $D = \top$.

1, 3 By the precondition, $L\sigma$ is undefined in Γ (3). Hence, adding $L\sigma$ does not make Γ inconsistent (1).

2, 4 Trivially fulfilled by hypothesis.

5 Since $D = \top$, the rule is trivially satisfied.

6 For all literals $L'\sigma' \in \Gamma$, this holds by hypothesis. For $L\sigma$ this follows directly from the preconditions of the rule.

Propagate. Assume Propagate is applicable to $(\Gamma; N; U; \beta; k; D)$, yielding a resulting state $(\Gamma, L\sigma^{(C_0 \vee L)\delta \cdot \sigma}; N; U; \beta; k; D)$. Then, there is a $C \vee L \in (N \cup U)$ such that $C = C_0 \vee C_1$, $C_1\sigma = L\sigma \vee \cdots \vee L\sigma$, $C_0\sigma$ does not contain $L\sigma$, δ is the mgu of the literals in C_1 and L, $(C \vee L)\sigma$ is ground, $(C \vee L)\sigma \prec_\beta \{\beta\}$, $C_0\sigma$ is false under Γ, and $L\sigma$ is undefined in Γ. Also, there can be no active conflict, i.e. $D = \top$.

1, 3 By the precondition, $L\sigma$ is undefined in Γ (3). Hence, adding $L\sigma^{(C_0 \vee L)\delta \cdot \sigma}$ does not make Γ inconsistent (1).

2 Consider any decomposition $\Gamma, L\sigma^{(C_0 \vee L)\delta \cdot \sigma} = \Gamma_1, L'\sigma' {}^{C_0' \vee L' \cdot \sigma'}, \Gamma_2$. In the case of $L'\sigma \neq L\sigma$, we can apply the hypothesis for the state $(\Gamma; N; U; \beta; k; D)$. Hence, only the case $\Gamma_1 = \Gamma$, $L'\sigma' = L\sigma$, and $C_0'\sigma = C_0\sigma$ is left to prove.

First, note that $C_0\sigma$ is false under $\Gamma_1 = \Gamma$ by the preconditions. Also, $L\sigma$ must be undefined in Γ by the preconditions. Lastly, it needs to be shown that $N \cup U \models (C_0 \vee L)\delta$. Clearly, since $C \vee L \in (N \cup U)$, it holds that $N \cup U \models C \vee L$. Since $C = C_0 \vee C_1$ and $C_1\sigma = L\sigma \vee \cdots \vee L\sigma$ it follows from the soundness of first-order factorization that $C \models (C_0 \vee L)$ and by this $N \cup U \models C_0 \vee L$.

4 Follows trivially from the induction hypothesis.

5 Since $D = \top$, this rule is trivially satisfied.

6 For all literals $L'\sigma' \in \Gamma$, this holds by hypothesis. For $L\sigma$, consider the precondition that $(C \vee L)\sigma \prec_\beta \{\beta\}$. By the definition of the multiset extension of $\prec_\beta$, it follows that $L\sigma \prec_\beta \beta$ must hold as well.

Conflict. Assume Conflict is applicable to $(\Gamma; N; U; \beta; k; D)$, yielding a resulting state $(\Gamma; N; U; \beta; k; C \cdot \sigma)$. Then, there is a $C \in (N \cup U)$ such that $C\sigma$ is false in Γ for a grounding σ.

1-3 Trivially fulfilled by hypothesis, as the trail Γ is not modified.

4 Follows trivially from the induction hypothesis, as neither N nor U are modified.

5 It holds that $D = C \cdot \sigma$. By the preconditions of Conflict, $C\sigma$ must be false under Γ. Furthermore, since $C \in (N \cup U)$ it holds that $N \cup U \models C$. Since $N \models U$ by soundness (4), is also holds that $N \models C$. Lastly, it remains to show that $\mathrm{grd}^{\prec_\beta \beta}(N) \models C\sigma$. By soundness (6), we know that for all literals $L\mu \in \Gamma$ it holds that $L\mu \prec_\beta \beta$. Since $C\sigma$ is false in Γ, it must hold that all literals in $C\sigma$ are also $\prec_\beta \beta$. Combined with $N \models C$, this yields that $\mathrm{grd}^{\prec_\beta \beta}(N) \models C\sigma$.

6 Fulfilled by the hypothesis, since no literal is added to Γ.

Skip. Assume Skip is applicable to $(\Gamma = \Gamma', L; N; U; \beta; k; D \cdot \sigma)$, yielding a resulting state $(\Gamma'; N; U; \beta; k - i; D \cdot \sigma)$. By the preconditions of skip, it must hold that $\mathrm{comp}(L)$ does not occur in $D\sigma$, and if L is a decision literal then $i = 1$ else $i = 0$.

1-3, 6 Directly fulfilled by hypothesis, as all prefixes of Γ still fulfil all properties. In particular, this holds for the prefix Γ' of Γ.

4 Follows trivially from the induction hypothesis, as U is not modified.

5 After the application of Skip, $D \cdot \sigma$ is the current conflict. Since D is not modified, $N \models D$ and $\mathrm{grd}^{\prec_\beta \beta}(N) \models D\sigma$ by hypothesis. It is left to show that $D\sigma$ is false under the resulting Γ', given the assumption that $D\sigma$ is false under Γ. However, since $\mathrm{comp}(L) \notin D\sigma$, this is trivially fulfilled, as the removal of $\mathrm{comp}(L)$ from the trail Γ cannot make $D\sigma$ undefined. Hence, $D\sigma$ must be false under Γ' as well.

Factorize. Assume Factorize is applicable to $(\Gamma; N; U; \beta; k; (D \vee L \vee L') \cdot \sigma)$, yielding a resulting state $(\Gamma; N; U; \beta; k; (D \vee L)\eta \cdot \sigma)$. Then, $L\sigma = L'\sigma$ and $\eta = \mathrm{mgu}(L, L')$.

1-3, 6 Trivially fulfilled by hypothesis, as the trail Γ is not modified.

4 Follows trivially from the induction hypothesis, as U is not modified.

5 After the application of Factorize, $(D \vee L)\eta \cdot \sigma$ is the current conflict. By the hypothesis $N \models (D \vee L \vee L')$. From the preconditions of Factorize, $L\sigma = L'\sigma$ and $\eta = \mathrm{mgu}(L, L')$. Thus, $(D \vee L \vee L')\eta$ is an instance of $(D \vee L \vee L')$ and $N \models (D \vee L \vee L')\eta$. Since $L\eta = L'\eta$, $(D \vee L \vee L')\eta \models (D \vee L')\eta$. Thus, $N \models (D \vee L)\eta$. By the preconditions, $\mathrm{grd}^{\prec_\beta \beta}(N) \models \mathrm{grd}^{\prec_\beta \beta}((L \vee L \vee L')\sigma)$. Hence, $(D \vee L \vee L')\sigma \prec_\beta \{\beta\}$. Thus, $(D \vee L)\eta\sigma = (D \vee L)\sigma \prec_\beta \{\beta\}$. From this, it follows that $\mathrm{grd}^{\prec_\beta \beta}(N) \models \mathrm{grd}^{\prec_\beta \beta}((D \vee L)\sigma)$.

Furthermore, $(D \vee L)\eta\sigma$ is false under Γ, since $(D \vee L)\eta\sigma = (D \vee L)\sigma$ by the definition of an mgu, and $(D \vee L \vee L')\sigma$ is already false under Γ.

Resolve. Assume the rule Resolve is applicable to an SCL state of the shape $(\Gamma = \Gamma', L\delta^{(C \vee L) \cdot \delta}; N; U; \beta; k; (D \vee L') \cdot \sigma)$, yielding a resulting state $(\Gamma; N; U; \beta; k; (D \vee C)\eta \cdot \sigma\delta)$. By the preconditions of Resolve, it holds that $L\delta = \mathrm{comp}(L'\sigma)$ and $\eta = \mathrm{mgu}(L, \mathrm{comp}(L'))$.

1-3, 6 Trivially fulfilled by hypothesis, as the trail Γ is not modified.

4 Follows trivially from the induction hypothesis, as U is not modified.

5 After the application of Resolve, $(D \vee C)\eta \cdot \sigma\delta$ is the current conflict.

By the hypothesis, $(D \vee L')\sigma$ is false under Γ. In particular, $D\sigma$ is false under Γ. By soundness (2), we know that $C\delta$ must be false under Γ as well. Hence, $(D \vee L)\eta\sigma\delta$ is false under Γ.

Furthermore, by the hypothesis, $N \models (D \vee L')$. Since $(D \vee L')\eta$ is an instance of $(D \vee L')$, it holds that $N \models (D \vee L')\eta$. Furthermore, by soundness (2) we know that $N \cup U \models (C \vee L)$ and by soundness (4) this implies that $N \models (C \vee L)$. With similar argumentation, also $N \models (C \vee L)\eta$. By the soundness of resolution, this implies $N \models (D \vee C)\eta$.

Lastly, since $(D \vee L')\sigma$ is false in Γ, all occuring literals in $\{(D \vee L')\sigma\} \prec_\beta \{\beta\}$. With similar argumentation, $\{(C \vee L)\delta\} \prec_\beta \{\beta\}$. Hence, in particular, $(D \vee C)\eta\sigma\delta \prec_\beta \{\beta\}$ and, thus, $\mathrm{grd}^{\prec_\beta \beta}(N) \models \mathrm{grd}^{\prec_\beta \beta}((D \vee C)\eta\sigma\delta)$.

Backtrack. Assume the rule Backtrack is applicable to a SCL state of shape $(\Gamma = \Gamma_0, K, \Gamma_1; N; U; \beta; k; (D \vee L) \cdot \sigma)$, yielding the resulting SCL state $(\Gamma_0, K; N; U \cup \{D \vee L\}; \beta; k'; \top)$.

1-3, 6 Directly fulfilled by hypothesis, as all prefixes of Γ still fulfil all properties. In particular, this holds for the prefix Γ_0, K of Γ.

4 By the hypothesis, we know that $N \models U$. By soundness (5) we know that $N \models (D \vee L)$. Overall, $N \models U \cup \{D \vee L\}$

5 Since after an application of Backtrack the conflict is resolved, i.e. $D = \top$, the rules are trivially satisfied.

□

Corollary 3.16.11. The rules of SCL are sound, hence SCL starting with an initial state is sound.

Proof. Follows by induction over the length of the run. The base case is handled by Lemma 3.16.9, the induction step is contained in Theorem 3.16.10. □

Next we introduce reasonable and regular runs. As an overall goal, we will show that regular runs always generate non-redundant clauses but do not require exhaustive propagation.

Definition 3.16.12 (Reasonable Runs)**.** A sequence of SCL rule applications is called a *reasonable run* if the rule Decide does not enable an immediate application of rule Conflict.

Definition 3.16.13 (Regular Runs)**.** A sequence of SCL rule applications is called a *regular run* if it is a reasonable run and the rule Conflict has precedence over all other rules.

Theorem 3.16.14 (Correct Termination)**.** If in a regular run no rules are applicable to a state $(\Gamma; N; U; \beta; k; D)$ then either $D = \bot$ and N is unsatisfiable or $D = \top$ and $\mathrm{grd}(N)^{\prec_\beta \beta}$ is satisfiable and $\Gamma \models \mathrm{grd}(N)^{\prec_\beta \beta}$.

Proof. Consider a state $(\Gamma; N; U; \beta; k; D)$. Then, D can have one of the following shapes:

(Case $D = \top$) If $D = \top$, then there is no active conflict. Assume there are no undefined ground literals $L \prec_\beta \beta$ for $L \in C$, $C \in N \cup U$ in Γ. Now, either $\Gamma \models \mathrm{grd}^{\prec_\beta \beta}(N)$ and thus Γ is already a partial model for N w.r.t. $\prec_\beta$ and β. Otherwise, if $\Gamma \not\models \mathrm{grd}^{\prec_\beta \beta}(N)$ but all literals are defined, there must be a false clause $C \in \mathrm{grd}^{\prec_\beta \beta}(N)$ which can be chosen as a Conflict instance.

If there is at least one undefined ground literal $L \prec_\beta \beta$ occuring in $N \cup U$, one of the trail building rules Propagate, Decide, or Conflict are applicable. Decide on the undefined ground literal L is, by the preconditions of the rule, in such a case always possible. The application of Decide can, however, be restricted by reasonability or regularity.

If Decide on L is not applicable by reasonability, then Γ, L^{k+1} must lead to a direct application of Conflict. Thus, there is a clause $D \in N \cup U$ such that $D\sigma$ is false under Γ, L^{k+1}. If $D\sigma$ is already false under Γ, then Conflict is applicable. Otherwise, D has the shape $D_0 \vee D_1$ where D_0 is false under Γ, and $D_1\sigma = \mathrm{comp}(L) \vee \cdots \vee \mathrm{comp}(L)$. Since D_0 is false under Γ, also $D_0 \prec_\beta \{\beta\}$ and

since $L \prec_\beta \beta$ it holds that $D_0 \vee D_1 \prec_\beta \{\beta\}$ by the definition of our multiset extension. Hence, Propagate can be applied.

If Decide is not applicable by regularity, Conflict must be applicable, since regularity only priorizes the Conflict rule application.

(Case $D = C \cdot \sigma$) If $D = C \cdot \sigma$, then there is an active conflict which needs to be resolved. In this case, one of the rules Resolve, Skip, Factorize or Backtrack are applicable.

First, consider the case of $\Gamma = \varepsilon$. By soundness, $C\sigma$ must be false under Γ. However, the only false clause under ε is $\bot$. In this case, $D = \bot$ and by soundness, $N \models \bot$. Hence, N is unsatisfiable. In the other case, there is at least one literal on the trail. We split $\Gamma = \Gamma', L$ and distinguish the shape of L:

- Consider the case that L was propagated, i.e. is of shape $L^{C \cdot \delta}$ for a clause $C \in N \cup U$. Then, either Resolve or Skip are applicable. In the case that $\mathrm{comp}(L)$ occurs in $C\sigma$, Resolve is applicable. If $\mathrm{comp}(L) \notin C\sigma$, Skip is applicable.

- Consider the case that L is a decision literal, i.e. is of shape L^i for a numerical level i. Then, one of the rules Skip, Backtrack or Factorize are applicable.

  If $\mathrm{comp}(L)$ does not occur in $C\sigma$, then Skip can be applied. Backtrack can be applied in all other cases if $C = (C' \vee \mathrm{comp}(L))$, where C' is of level $i' < k$. Note that for Backtrack there must be a level j that is backtracked to. This level j always exists if all other preconditions are met. Hence, if Skip is not applicable, C is of the shape $C' \vee \mathrm{comp}(L)$. If C' is of level k, then Factorize can be applied instead, as C' must contain another instance of $\mathrm{comp}(L)$. Otherwise, C' is of level $i' < k$ and Backtrack can be applied.

For a state $(\Gamma; N; U; \beta; k; D)$ where $D \notin \{\top, \bot\}$, one of the rules Resolve, Skip, Factorize or Backtrack is applicable. If the top level literal is a propagated literal then either Resolve or Skip are applicable. If the top level literal is a decision then one of the rules Skip, Backtrack, or Factorize is applicable. In the case $D = \top$ and Decide is not applicable by regularity, Propagate can always be applied instead. If $D = \top$ and all Propagate, Decide, and Conflict are not applicable it means that there are no undefined ground literals $L \prec_\beta \beta$ in Γ, so $\Gamma \models \mathrm{grd}(N)^{\prec_\beta \beta}$.

□

Lemma 3.16.15 (Resolve in regular runs)**.** Consider the derivation of a conflict state $(\Gamma, L; N; U; \beta; k; \top) \Rightarrow_{\text{Conflict}} (\Gamma, L; N; U; \beta; k; D)$. In a regular run, during conflict resolution L is not a decision literal and at least the literal L is resolved.

Proof. In a reasonable run, if the rule Decide has produced the SCL state $(\Gamma, L; N; U; \beta; k; \top)$, the rule Conflict is not immediately applicable, so L must be a propagated literal. In case the rule Backtrack produced the state

$(\Gamma, L; N; U; \beta; k; \top)$ there is the sequence of rule applications

$$\begin{aligned}&(\Gamma, L, L', \Gamma_1, K^{k+1}, \Gamma_2, \mathrm{comp}(L\sigma)^{k'}; N; U'; \beta; k'; (D \vee L'') \cdot \sigma)\\ \Rightarrow_{\mathrm{SCL}}^{\mathrm{Backtrack}} &(\Gamma, L; N; U' \cup (D \vee L''); \beta; k; \top)\end{aligned}$$

Then, by the definition of Backtrack, the newly learned clause $(D \vee L'')$ cannot be false with respect to Γ, L. Thus, Conflict is not applicable to $(D \vee L'')$. In summary, L must be a propagated literal.

Backtrack is not directly applicable to $(\Gamma, L; N; U; \beta; k; D)$, as it requires L to be a decision literal. Furthermore, L must occur in the conflict clause D. Otherwise, Conflict could have been applied earlier to $(\Gamma; N; U; \beta; k; \top)$, contradicting regularity. Hence, Skip is not applicable to our state. Overall, only Factorize and Resolve can possibly be applied to our state. After an application of Factorize, the two invariants still hold: First, the trail is not modified. Second, L must still occur in the conflict clause D, as Factorize cannot remove all instances of L from D. Hence, Factorize cannot enable any of the rules Skip or Backtrack. Following from that, at least one application of Resolve must take place in conflict resolution. □

Definition 3.16.16 (State Induced Ordering)**.** Let $(L_1, L_2, \ldots, L_n; N; U; \beta; k; D)$ be a sound state of SCL. The trail induces a total well-founded strict order on the defined literals by

$$L_1 \prec_\Gamma \mathrm{comp}(L_1) \prec_\Gamma L_2 \prec_\Gamma \mathrm{comp}(L_2) \prec_\Gamma \cdots \prec_\Gamma L_n \prec_\Gamma \mathrm{comp}(L_n).$$

We extend $\prec_\Gamma$ to a strict total order on all literals where all undefined literals are larger than $\mathrm{comp}(L_n)$. We also extend $\prec_\Gamma$ to a strict total order on ground clauses by multiset extension and also on multisets of ground clauses and overload $\prec_\Gamma$ for all these cases. With $\preceq_\Gamma$ we denote the reflexive closure of $\prec_\Gamma$.

Theorem 3.16.17 (Learned Clauses in Regular Runs)**.** Let $(\Gamma; N; U; \beta; k; C_0 \cdot \sigma_0)$ be the state resulting from the application of Conflict in a regular run and let C be the clause learned at the end of the conflict resolution, then C is not redundant with respect to $N \cup U$ and $\prec_\Gamma$.

Proof. Consider the following fragment of a derivation learning a clause:
$\Rightarrow_{\mathrm{SCL}}^{\mathrm{Conflict}} (\Gamma; N; U; \beta; k; C_0 \cdot \sigma_0) \Rightarrow_{\mathrm{SCL}}^{\{\mathrm{Skip,\ Fact.,\ Res.}\}^*} (\Gamma'; N; U; \beta; k; C \cdot \sigma) \Rightarrow_{\mathrm{SCL}}^{\mathrm{Backtrack}}$.
Consider the following fragment of a derivation learning a clause:

$$\begin{aligned}&\Rightarrow_{\mathrm{SCL}}^{\mathrm{Conflict}} && (\Gamma; N; U; \beta; k; C_0 \cdot \sigma_0)\\ &\Rightarrow_{\mathrm{SCL}}^{\{\mathrm{Skip,\ Fact.,\ Res.}\}^*} && (\Gamma'; N; U; \beta; k; C \cdot \sigma)\\ &\Rightarrow_{\mathrm{SCL}}^{\mathrm{Backtrack}} &&\end{aligned}$$

By soundness $N \cup U \models C$ and $C\sigma$ is false under both Γ and Γ'. We prove that $C\sigma$ is non-redundant to $N \cup U$ with respect to $\prec_\Gamma$.

Assume there is an $S \subseteq \mathrm{grd}(N \cup U)^{\preceq_\Gamma C\sigma}$ such that $S \models C\sigma$. There must be a clause $D \in S$ false under Γ, since all clauses in S have a defined truth value

(as all undefined literals are greater in $\prec_\Gamma$ than all defined literals) and if $\Gamma \models S$ then $\Gamma \models C\sigma$ by transitivity of entailment, a contradiction.

By regularity, Γ must be of the shape $\Gamma = \Gamma'', L\delta^{C \vee L \cdot \delta}$, since no application of Decide can lead to an application of the rule Conflict. Thus, the last applied rule must have been Propagate. Furthermore, by Lemma 3.16.15, Resolve must have resolved at least the rightmost literal $L\delta$ from Γ. Thus, $L\delta \notin C\sigma$ and $\mathrm{comp}(L\delta) \notin C\sigma$.

Since $D \prec_\Gamma C\sigma$, neither $L\delta$ nor $\mathrm{comp}(L\delta)$ may occur in D. However, this is a contradiction, since D is then already false under Γ'' and, thus, must have been chosen as a Conflict instance earlier in a regular run. □

Of course, in a regular run, the ordering of foreground literals on the trail will change, i.e., the ordering of Definition 3.16.16 will change as well. Thus the non-redundancy property of Lemma 3.16.17 reflects the situation at the time of creation of the learned clause. A non-redundancy property holding for an overall run must be invariant against changes on the ordering. However, the ordering of Definition 3.16.16 also entails a fixed subset ordering $\prec_\subseteq$ that is invariant against changes on the overall ordering. This means that our dynamic ordering entails non-redundancy criteria based on subset relations including forward subsumption. From an implementation perspective, this means that learned clauses need not to be tested for forward redundancy. Current resolution or superposition based provers spent a reasonable portion of their time in testing forward redundancy of newly generated clauses. In addition, also tests for backward reduction can be restricted knowing that learned clauses are not redundant.

Theorem 3.16.18 (BS Non-Redundancy is NEXPTIME-Complete)**.** Deciding non-redundancy of a BS clause C with respect to a finite BS clause set $N^{\preceq C}$ is NEXPTIME-Complete.

Proof. We only show hardness, because containment of the problem in NEXPTIME is obvious. To this end, let $N = \{C_1, \ldots, C_n\}$ be an arbitrary, finite BS clause set. We consider an LPO ordering $\prec_{\mathrm{LPO}}$. Next, we add a fresh predicate P of arity zero, where P is $\prec_{\mathrm{LPO}}$ larger than any clause in N. Now, in the finite BS clause set $N \cup \{P\}$ the clause P is redundant iff N is unsatisfiable. □

Theorem 3.16.19 (Termination)**.** Any regular run of $\Rightarrow_{\mathrm{SCL}}$ terminates.

Proof. Any infinite run learns infinitely many clauses. Firstly, for a regular run, by Theorem 3.16.17, all learned clauses are non-redundant. By Remark 3.16.4, those clauses are also non-redundant under the fixed subset ordering $\prec_\subseteq$, which is well-founded. Due to the restriction of all clauses to be smaller than $\{\beta\}$, the overall number of non-redundant ground clauses is finite. So there is no infinite regular run. □

Theorem 3.16.20 (SCL Refutational Completeness)**.** If N is unsatisfiable, such that some finite $N' \subseteq \mathrm{grd}(N)$ is unsatisfiable and β is $\prec_\beta$ larger than all literals in N' then any regular run from $(\epsilon; N; \emptyset; \beta; 0; \top)$ of SCL derives $\bot$.

Proof. By Theorem 3.16.19 and Theorem 3.16.14. □

Obviously, given some unsatisfiable clause set N there is no way to efiiciently compute some β such that $ground(N)^{\prec_\beta}$ is unsatisfiable. Therefore, in an implementation, the below rule Grow is needed to eventually provide a semi-decision procedure.

Grow $(\Gamma; N; U; \beta; k; \top) \Rightarrow_{\mathrm{SCL}} (\epsilon; N; U; \beta'; 0; \top)$
provided $\Gamma \models \mathrm{grd}(N)^{\prec_\beta}$ and $\beta \prec_\beta \beta'$

Theorem 3.16.21 (SCL decides the BS fragment)**.** SCL restricted to regular runs decides satisfiability of a BS clause set if β is set appropriately.

Proof. Let B be the set of constants in the BS clause set N. Then define $\prec_\beta$ and β such that $L \prec_\beta \beta$ for all $L \in \mathrm{grd}^{\prec_\beta \beta}(N)$. Following the proof of Theorem 3.16.19, any SCL regular run will terminate on a BS clause set. □

Example 3.16.22 (Comparing Proof Length Depending on Unit Clause Propagation)**.** Proofs generated without full propagation can be exponentially shorter than proofs generated by exhaustive propagation. Consider the simple BS clause set over constants $\Omega = \{a, b\}$

$$N = \{R(x_1, \ldots, x_n, a, b), P \vee Q, P \vee \neg Q, \neg P \vee Q, \neg P \vee \neg Q\}$$

A run without exhaustive propagation can ignore generating the 2^n different ground instances of $R(x_1, \ldots, x_n, a, b)$ starting with initial set $B = \{a, b\}$. Instead it refutes the propositional part of N in the usual CDCL style by starting with a decision on P or Q. For the example it is obvious that the instances of $R(x_1, \ldots, x_n, a, b)$ can be ignored, but in general it is not.

Consider another example, taken from [**?**], where exhaustive propagation leads to exponentially longer proofs compared to the shortest resolution proof.

Example 3.16.23 (Comparing Proof Length Depending on Clause Propagation)**.** Let i be a positive integer and consider the clause set N^i with one predicate P of arity i consisting of the following clauses, where we write $\bar{x}, \bar{0}$ and $\bar{1}$ to denote sequences of the appropriate length of variables and constants to meet the arity of P:

$$P(\bar{0}) \qquad \neg P(\bar{1})$$

and i clauses of the form $\neg P(\bar{x}, 0, \bar{1}) \vee P(\bar{x}, 1, \bar{0})$
where the length of $\bar{1}$ varies between 0 and $i - 1$. The example encodes an i-bit counter. An SCL run with exhaustive propagation on this clause set finds a conflict after $O(2^i)$ propagations without any application of Decide.

For the instance $i = 4$ we get the clauses of N^4:

$$N^4 = \{ \begin{array}{l} 1 : P(0,0,0,0) \\ 2 : \neg P(x_1,x_2,x_3,0) \vee P(x_1,x_2,x_3,1) \\ 3 : \neg P(x_1,x_2,0,1) \vee P(x_1,x_2,1,0) \\ 4 : \neg P(x_1,0,1,1) \vee P(x_1,1,0,0) \\ 5 : \neg P(0,1,1,1) \vee P(1,0,0,0) \\ 6 : \neg P(1,1,1,1) \end{array} \}$$

For this clause set an SCL all unit clauses from $P(0,0,0,0)$ to $P(1,1,1,1)$ via 2^4 applications of Propagate, then finds a conflict with clause 6 and then uses 2^4 times Resolve to end up in $\bot$.

Instead a short resolution refutation can be obtained by

2.2 Res 3.1	$7 : \neg P(x_1,x_2,0,0) \vee P(x_1,x_2,1,0)$
7.2 Res 2.1	$8 : \neg P(x_1,x_2,0,0) \vee P(x_1,x_2,1,1)$
8.2 Res 4.1	$9 : \neg P(x_1,0,0,0) \vee P(x_1,1,0,0)$
9.2 Res 8.1	$10 : \neg P(x_1,0,0,0) \vee P(x_1,1,1,1)$
10.2 Res 5.1	$11 : \neg P(0,0,0,0) \vee P(1,0,0,0)$
11.2 Res 10.1	$12 : \neg P(0,0,0,0) \vee P(1,1,1,1)$
12.1 Res 6.1	$13 : \bot$

In general, $O(2i)$ many resolution steps are sufficient to refute N^i. This derivation can be simulated by SCL if exhaustive propagation is not used. For example, the first resolution step between clauses 2.2 and 3.1 can be simulated by first deciding $P(1,1,0,0)$ and $\neg P(1,1,1,0)$ yielding the state

$$([P(1,1,0,0)^1, \neg P(1,1,1,0)^2]; N; \emptyset; \{0,1\}; 2; \top)$$

now we can propagate using $\neg P(1,1,1,0)^2$ with clause 3

$$([P(1,1,0,0)^1, \neg P(1,1,1,0)^2, \neg P(1,1,0,1)^{\neg P(x_1,x_2,0,1) \vee P(x_1,x_2,1,0) \cdot \{x_1 \mapsto 1, x_2 \mapsto 1\}}]; N; \emptyset; \{0,1\}; 2; \top)$$

and then get a conflict with clause 2 by closure $(\neg P(x_1,x_2,x_3,0) \vee P(x_1,x_2,x_3,1)) \cdot \{x_1 \mapsto 1, x_2 \mapsto 1, x_3 \mapsto 0\}$. Next we apply Conflict and Resolve to the rightmost propagated literal and get

$$([P(1,1,0,0)^1, \neg P(1,1,1,0)^2]; N; \emptyset; \{0,1\}; 2; (\neg P(x_1,x_2,0,0) \vee P(x_1,x_2,1,0)) \cdot \{x_1 \mapsto 1, x_2 \mapsto 1\}).$$

Finally Backtrack is applicable resulting in

$$([P(1,1,0,0)^1, P(1,1,1,0)^{(\neg P(x_1,x_2,0,0) \vee P(x_1,x_2,1,0)) \cdot \{x_1 \mapsto 1, x_2 \mapsto 1\}}]; N; \{\neg P(x_1,x_2,0,0) \vee P(x_1,x_2,1,0)\}; \{0,$$

However, to continue with the proof we need also need a Restart rule, which is anyway needed for completeness to get out of stuck states.