SUP(T) Decides the Ground Case

If the clause set N, not yet abstracted, of a hierarchic specification $\mathcal{H}$ is ground, then an instance of SUP(T) decides unsatisfiability of N, provided $\mathcal{T}^B$ enables a decision procedure for the applicability of Constraint Refutation.

An immediate application of SUP(T) to a ground clause set N does not yield a decision procedure, because N may not be sufficiently complete and hence SUP(T) may not be complete, and, SUP(T) does not necessarily terminate on N without further refinements.

Sufficient Completion $N \uplus \{\Lambda \parallel C[f(t_1, \ldots, t_n)]_{p_1,\ldots,p_n}\} \;\Rightarrow_{\mathrm{SUF}}$
$N \cup \{\Lambda \parallel C[b]_{p_1,\ldots,p_n},\; f(t_1, \ldots, t_n) \approx b\}$

provided f is a Σ^F function symbol ranging into a background theory sort, no t_i contains a Σ^F function symbol ranging into a background theory sort, and b is a fresh parameter from Σ^B, $p_1, \ldots, p_n$ are all positions of $f(t_1, \ldots, t_n)$ in C

For example, the ground clause

$$\neg P(f(h(1 + g(a)))) \lor f(h(1)) + g(a) \geq 0 \lor P(g(a)) \lor Q(g(a))$$

is replaced by the clauses

$$\begin{aligned}
&\neg P(b_3) \lor b_2 + b_1 \geq 0 \lor P(b_1) \lor Q(b_1)\\
&g(a) \approx b_1\\
&f(h(1)) \approx b_2\\
&f(h(1 + b_1)) \approx b_3
\end{aligned}$$

where $\mathcal{T}^B$ is LRA, a is not of sort $\mathbb{Q}$, h does not range into $\mathbb{Q}$, and the b_i are fresh parameters from LRA.

Next the clauses are abstracted resulting in the clause set

$$
\begin{array}{l}
y = b_1, x = b_3, b_2 + b_1 < 0 \parallel \neg P(x) \lor P(y) \lor Q(y) \\
y = b_1 \parallel g(a) \approx y \\
z = b_2, w = 1 \parallel f(h(w)) \approx z \\
u = 1 + b_1, v = b_3 \parallel f(h(u)) \approx v
\end{array}
$$

where now all introduced variables are equal in the constraint to a background theory ground term. The resulting clauses now have the property that all variables are variables of a background theory sort and that for all background variables $x \in C$ of some clause $\Lambda \parallel C$ there is a an atom $x = t \in \Lambda$ where t is a ground base term. In addtion, the only variable occurrences in Λ are equations $x = t$ for some ground term t.

The additional parameters b_i moved to the background theory part destroy compactness of LRA. However, compactness is not needed here, because I will eventually show termination of superposition on completed and abstracted clause sets. In order to show termination, clauses must not become arbitrarily long and terms must not become aribtrarily deep in the generated clauses.

The next step is to prevent variable chains. If in some clause $\Lambda \parallel C$ a variable x occurs several times in C, then in the context of sufficiently completed and abstracted ground clauses fresh variables y are introduced and the assignment $x = t \in \Lambda$ is copied for all y and added to Λ. For the first clause of the running example the result is

$$z = b_1, y = b_1, x = b_3, b_2 + b_1 < 0 \parallel \neg P(x) \vee P(y) \vee Q(z).$$

Single Free Vars $N \uplus \{\Lambda, x = t \parallel C[x]_{p_1,\dots,p_n}\} \Rightarrow_{\text{SVS}}$
$N \cup \{\Lambda, x = t, y_2 = t, \dots, y_{n-1} = t \parallel C[y_1/p_2, \dots, y_{n-1}/p_n]\}$
provided the y_i are fresh and $p_1, \dots, p_n$ are all posisitions of occurences of variables in C

Single Theo Vars $N \uplus \{\Lambda, x = t, x = s \parallel C\} \Rightarrow_{\text{SVS}}$
$N \cup \{\Lambda, x = t, t = s \parallel C\}$

Pure Theo Vars $N \uplus \{\Lambda, x = t \parallel C\} \Rightarrow_{\text{SVS}} N \cup \{\Lambda \parallel C\}$
provided $x \not\in \text{vars}(C)$, $x \not\in \text{vars}(\Lambda)$

A result of the abstraction or application of a superposition inference can also be variable equations on the free side, i.e., equations $x \approx y$ or $x \not\approx y$ occuring in the free part C of a clause $\Lambda \parallel C$. By the following two rules, such equations can be eliminated as well.

No VarEQ Pos $\quad N \uplus \{\Lambda, x = t, y = s \parallel C \vee x \approx y\} \Rightarrow_{\text{NVQ}}$
$N \cup \{\Lambda, s \neq t \parallel C\}$.

provided x, y do not occur in C

No VarEQ Neg $\quad N \uplus \{\Lambda, x = t, y = s \parallel C \vee x \not\approx y\} \Rightarrow_{\text{NVQ}}$
$N \cup \{\Lambda, s = t \parallel C\}$.

provided x, y do not occur in C

8.4.1 Lemma (Finite Clause Variations)

Let M be a finite set of ground literals of the background theory, T_M a finite set of ground terms of the background theory, and $k \in \mathbb{N}$ fixed. Then there are only finitely many non-redundat clauses $\Lambda \parallel C$ where the signature for C is finite and fixed if:

(1) any variable occurs at most once in $\Lambda \parallel C$,

(2) $\Lambda = \{x_1 = t_1, \ldots, x_n = t_n\} \uplus \Lambda'$ where $\Lambda' \subseteq M$, $t_i \in T_M$, all x_i are different and $\{x_1, \ldots, x_n\} = \mathrm{vars}(C)$, and

(3) $|\mathrm{atom}(L)| \leq k$ for all $L \in C$.

8.4.2 Definition ($\succ_{lpo}^{F}$)

For some term t let $|t|^F$ be the number of function symbols from Σ^F contained in t. Then $\succ_{lpo}^{F}$ is defined by $t \succ_{lpo}^{F} s$ iff

1. $|t|^F > |s|^F$ or
2. $|t|^F = |s|^F$ and $t \succ_{lpo} s$

8.4.3 Proposition (Reduction Ordering $\succ_{lpo}^{F}$)

With respect to simple substitions and terms only containing base variables $\succ_{lpo}^{F}$ is a reduction ordering.

8.4.4 Lemma (SUP(T) Termination)

SUP(T) terminates on any completed and abstracted ground clause set with respect to the ordering $\succ_{lpo}^{F}$, exhaustive $\Rightarrow_{\mathrm{SVS}}$, $\Rightarrow_{\mathrm{NVQ}}$ reduction on generated clauses, and subsumption and condensation.

8.4.5 Theorem (Decision Procedure)

SUP(T) is a decision procedure for a hierarchic specification $\mathcal{H} = (\mathcal{T}^H, \mathcal{T}^B)$ where N is ground and $\mathcal{T}^B$ provides a decision procedure for ground formulas with parameters.

Clauses where variables occur at most once can also be completely split into their components. For the running example, the clause $z = b_1, y = b_1, x = b_3, b_2 + b_1 < 0 \parallel \neg P(x) \vee P(y) \vee Q(z)$ can be split into the three clauses

$$\begin{array}{l} x = b_3, b_2 + b_1 < 0 \parallel \neg P(x) \\ y = b_1, b_2 + b_1 < 0 \parallel P(y) \\ z = b_1, b_2 + b_1 < 0 \parallel Q(z) \end{array}$$

where the inital clause set containing $z = b_1, y = b_1, x = b_3, b_2 + b_1 < 0 \parallel \neg P(x) \vee P(y) \vee Q(z)$ is unsatisfiable iff the three clause sets obtained by replacing the clause with one of the three split clauses $x = b_3, b_2 + b_1 < 0 \parallel \neg P(x)$, $y = b_1, b_2 + b_1 < 0 \parallel P(y)$, $z = b_1, b_2 + b_1 < 0 \parallel Q(z)$, respectively, is unsatisfiable.

Split $(\mathcal{N} \uplus \{N \uplus \{\Lambda_1, \Lambda_2 \parallel \Gamma_1, \Gamma_2 \rightarrow \Delta_1, \Delta_2\}\} \Rightarrow_{\text{SUP(T)}}$
$(\mathcal{N} \cup \{N \cup \{\Lambda_1 \parallel \Gamma_1 \rightarrow \Delta_1\}, N \cup \{\Lambda_2 \parallel \Gamma_2 \rightarrow \Delta_2\}\}$
if $\text{vars}(\Lambda_1 \parallel \Gamma_1 \rightarrow \Delta_1) \cap \text{vars}(\Lambda_2 \parallel \Gamma_2 \rightarrow \Delta_2) = \emptyset$ and $\Delta_1 \neq \emptyset$ and $\Delta_2 \neq \emptyset$