Automated Theorem Proving

Prof. Dr. Jasmin Blanchette, Yiming Xu, PhD,
Lydia Kondylidou, and Tanguy Bozec
based on questions by Dr. Uwe Waldmann

Winter Term 2025/26

For convenience, a handout is provided with the definitions of the main calculi and concepts covered in the course:

A. Summary of Main Definitions

A.1. Orderings

Let $\succ$ be a strict partial ordering on M; let $M' \subseteq M$.

$a \in M'$ is called *minimal in* M' if there is no $b \in M'$ with $a \succ b$.

$a \in M'$ is called *smallest in* M' if $b \succ a$ for all $b \in M' \setminus \{a\}$.

Analogously:

$a \in M'$ is called *maximal in* M' if there is no $b \in M'$ with $a \prec b$.

$a \in M'$ is called *largest in* M' if $b \prec a$ for all $b \in M' \setminus \{a\}$.

Moreover:

$a \in M'$ is called *strictly maximal in* M' if there is no $b \in M' - \{a\}$ with $a \preceq b$.

A.2. Multiset Orderings

Multiset Extensions Let $(M, \succ)$ be an abstract reduction system. The *multiset extension* of $\succ$ to multisets over M is defined by

$$\begin{aligned}
&S_1 \succ_{\text{mul}} S_2 \text{ if and only if}\\
&\qquad \text{there exist multisets } X \text{ and } Y \text{ over } M \text{ such that}\\
&\qquad\qquad \emptyset \neq X \subseteq S_1,\\
&\qquad\qquad S_2 = (S_1 - X) \cup Y,\\
&\qquad\qquad \forall y \in Y\ \exists x \in X\colon x \succ y
\end{aligned}$$

The *(Huet–Oppen) multiset extension* of $\succ$ to multisets over M is defined by

$$\begin{aligned}
&S_1 \succ_{\text{mul}}^{\text{HO}} S_2 \text{ if and only if}\\
&\quad S_1 \neq S_2 \text{ and}\\
&\qquad \forall m \in M\colon \big(S_2(m) > S_1(m)\\
&\qquad\qquad\qquad \Rightarrow\ \exists m' \in M\colon m' \succ m \text{ and } S_1(m') > S_2(m')\big)
\end{aligned}$$

A third way to characterize the multiset extension of a binary relation $\succ$ is to define it as the transitive closure of the relation $\succ_{\text{mul}}^{1}$ given by

$$\begin{aligned}
&S_1 \succ_{\text{mul}}^{1} S_2 \text{ if and only if}\\
&\qquad \text{there exists } x \in S_1 \text{ and a multiset } Y \text{ over } M \text{ such that}\\
&\qquad\qquad S_2 = (S_1 - \{x\}) \cup Y,\\
&\qquad\qquad \forall y \in Y\colon x \succ y
\end{aligned}$$

A.3. CNF Transformation for Propositional Logic

We describe a (naive) algorithm to convert a formula to CNF.

Apply the following rules as long as possible (modulo commutativity of $\wedge$ and $\vee$):

Step 1: Eliminate equivalences:

$$H[F \leftrightarrow G]_p \;\Rightarrow_{\text{CNF}}\; H[(F \rightarrow G) \wedge (G \rightarrow F)]_p$$

Step 2: Eliminate implications:

$$H[F \rightarrow G]_p \;\Rightarrow_{\text{CNF}}\; H[\neg F \vee G]_p$$

Step 3: Push negations inward:

$$\begin{aligned} H[\neg(F \vee G)]_p &\;\Rightarrow_{\text{CNF}}\; H[\neg F \wedge \neg G]_p \\ H[\neg(F \wedge G)]_p &\;\Rightarrow_{\text{CNF}}\; H[\neg F \vee \neg G]_p \end{aligned}$$

Step 4: Eliminate multiple negations:

$$H[\neg\neg F]_p \;\Rightarrow_{\text{CNF}}\; H[F]_p$$

Step 5: Push disjunctions inward:

$$H[(F \wedge F') \vee G]_p \;\Rightarrow_{\text{CNF}}\; H[(F \vee G) \wedge (F' \vee G)]_p$$

Step 6: Eliminate $\top$ and $\bot$:

$$\begin{aligned} H[F \wedge \top]_p &\;\Rightarrow_{\text{CNF}}\; H[F]_p \\ H[F \wedge \bot]_p &\;\Rightarrow_{\text{CNF}}\; H[\bot]_p \\ H[F \vee \top]_p &\;\Rightarrow_{\text{CNF}}\; H[\top]_p \\ H[F \vee \bot]_p &\;\Rightarrow_{\text{CNF}}\; H[F]_p \\ H[\neg\bot]_p &\;\Rightarrow_{\text{CNF}}\; H[\top]_p \\ H[\neg\top]_p &\;\Rightarrow_{\text{CNF}}\; H[\bot]_p \end{aligned}$$

A.4. DPLL

```
boolean DPLL(literal set $M$, clause set $N$) {
    if (all clauses in $N$ are true in $M$) return true;
    elsif (some clause in $N$ is false in $M$) return false;
    elsif ($N$ contains unit literal $P$) return DPLL($M \cup \{P\}$, $N$);
    elsif ($N$ contains unit literal $\neg P$) return DPLL($M \cup \{\neg P\}$, $N$);
    elsif ($N$ contains pure literal $P$) return DPLL($M \cup \{P\}$, $N$);
    elsif ($N$ contains pure literal $\neg P$) return DPLL($M \cup \{\neg P\}$, $N$);
    else {
        let $P$ be some undefined variable in $N$;
        if (DPLL($M \cup \{\neg P\}$, $N$)) return true;
        else return DPLL($M \cup \{P\}$, $N$);
    }
}
```

A.5. CNF Transformation for First-Order Logic

Prenex Normal Form Computing prenex normal form by the reduction system $\Rightarrow_P$:

$$\begin{array}{rcl}
H[(F \leftrightarrow G)]_p & \Rightarrow_P & H[(F \rightarrow G) \wedge (G \rightarrow F)]_p \\
H[\neg \mathsf{Q}x\, F]_p & \Rightarrow_P & H[\overline{\mathsf{Q}}x\, \neg F]_p \\
H[((\mathsf{Q}x\, F) \;\circ\; G)]_p & \Rightarrow_P & H[\mathsf{Q}y\, (F\{x \mapsto y\} \;\circ\; G)]_p, \\
 & & \quad \circ \in \{\wedge, \vee\} \\
H[((\mathsf{Q}x\, F) \rightarrow G)]_p & \Rightarrow_P & H[\overline{\mathsf{Q}}y\, (F\{x \mapsto y\} \rightarrow G)]_p, \\
H[(F \;\circ\; (\mathsf{Q}x\, G))]_p & \Rightarrow_P & H[\mathsf{Q}y\, (F \;\circ\; G\{x \mapsto y\})]_p, \\
 & & \quad \circ \in \{\wedge, \vee, \rightarrow\}
\end{array}$$

Here y is always assumed to be some fresh variable and $\overline{\mathsf{Q}}$ denotes the quantifier *dual* to Q, i.e., $\overline{\forall} = \exists$ and $\overline{\exists} = \forall$.

Skolemization Transformation $\Rightarrow_S$
(to be applied outermost, *not* in subformulas):

$$\forall x_1, \ldots, x_n \exists y\, F \quad \Rightarrow_S \quad \forall x_1, \ldots, x_n\, F\{y \mapsto f(x_1, \ldots, x_n)\}$$

where f/n is a new function symbol (*Skolem function*).

The Complete Picture

$$\begin{array}{lll}
F & \Rightarrow_P^* & \mathsf{Q}_1 y_1 \dots \mathsf{Q}_n y_n \, G \qquad\qquad (G \text{ quantifier-free}) \\
 & \Rightarrow_S^* & \forall x_1, \dots, x_m \, H \qquad (m \leq n,\ H \text{ quantifier-free}) \\
 & \Rightarrow_{CNF}^* & \underbrace{\underbrace{\forall x_1, \dots, x_m}_{\text{leave out}} \bigwedge_{i=1}^{k} \underbrace{\bigvee_{j=1}^{n_i} L_{ij}}_{\text{clauses } C_i}}_{F'}
\end{array}$$

$N = \{C_1, \dots, C_k\}$ is called the *clausal (normal) form* of F.
Note: The variables in the clauses are implicitly universally quantified.

A.6. Unification

Rule-Based Naive Standard Unification

$$\begin{array}{rcl}
t \doteq t, E & \Rightarrow_{SU} & E \\
f(s_1, \dots, s_n) \doteq f(t_1, \dots, t_n), E & \Rightarrow_{SU} & s_1 \doteq t_1, \dots, s_n \doteq t_n, E \\
f(\dots) \doteq g(\dots), E & \Rightarrow_{SU} & \bot \\
 & & \text{if } f \neq g \\
x \doteq t, E & \Rightarrow_{SU} & x \doteq t, E\{x \mapsto t\} \\
 & & \text{if } x \in \text{var}(E), x \notin \text{var}(t) \\
x \doteq t, E & \Rightarrow_{SU} & \bot \\
 & & \text{if } x \neq t, x \in \text{var}(t) \\
t \doteq x, E & \Rightarrow_{SU} & x \doteq t, E \\
 & & \text{if } t \notin X
\end{array}$$

If $E = \{x_1 \doteq u_1, \dots, x_k \doteq u_k\}$, with x_i pairwise distinct, $x_i \notin \text{var}(u_j)$, then E is called an (equational problem in) *solved form* representing the solution $\sigma_E = \{x_1 \mapsto u_1, \dots, x_k \mapsto u_k\}$.

Rule-Based Polynomial Unification

$$\begin{array}{rll}
t \doteq t, E & \Rightarrow_{PU} & E \\
f(s_1, \ldots, s_n) \doteq f(t_1, \ldots, t_n), E & \Rightarrow_{PU} & s_1 \doteq t_1, \ldots, s_n \doteq t_n, E \\
f(\ldots) \doteq g(\ldots), E & \Rightarrow_{PU} & \bot \\
& & \text{if } f \neq g \\
x \doteq y, E & \Rightarrow_{PU} & x \doteq y, E\{x \mapsto y\} \\
& & \text{if } x \in \mathrm{var}(E), x \neq y \\
x_1 \doteq t_1, \ldots, x_n \doteq t_n, E & \Rightarrow_{PU} & \bot \\
& & \text{if there are positions } p_i \text{ with} \\
& & t_i|_{p_i} = x_{i+1}, t_n|_{p_n} = x_1 \\
& & \text{and some } p_i \neq \varepsilon \\
x \doteq t, E & \Rightarrow_{PU} & \bot \\
& & \text{if } x \neq t, x \in \mathrm{var}(t) \\
t \doteq x, E & \Rightarrow_{PU} & x \doteq t, E \\
& & \text{if } t \notin X \\
x \doteq t, x \doteq s, E & \Rightarrow_{PU} & x \doteq t, t \doteq s, E \\
& & \text{if } t, s \notin X \text{ and } |t| \leq |s|
\end{array}$$

To obtain the unifier $\sigma_{E'}$, we have to sort the list of equality problems $x_i \doteq t_i$ in such a way that x_i does not occur in t_j for $j < i$, and then we have to compose the substitutions $\{x_1 \mapsto t_1\} \circ \cdots \circ \{x_k \mapsto t_k\}$.

A.7. Ordered Resolution with Selection

Ground Clause Orderings

1. We assume that $\succ$ is any fixed ordering on ground atoms that is *total* and *well-founded*. (There exist many such orderings, e.g., the length-based ordering on atoms when these are viewed as words over a suitable alphabet.)

2. Extend $\succ$ to an *ordering* $\succ_{\mathrm{L}}$ *on ground literals:*

$$\begin{array}{lcrl}
A & \succ_{\mathrm{L}} & B & \text{if } A \succ B \\
A & \succ_{\mathrm{L}} & \neg B & \text{if } A \succ B \\
\neg A & \succ_{\mathrm{L}} & B & \text{if } A \succ B \\
\neg A & \succ_{\mathrm{L}} & \neg B & \text{if } A \succ B \\
\neg A & \succ_{\mathrm{L}} & A &
\end{array}$$

3. Extend $\succ_{\mathrm{L}}$ to an *ordering* $\succ_{\mathrm{C}}$ *on ground clauses:*
   $\succ_{\mathrm{C}} = (\succ_{\mathrm{L}})_{\mathrm{mul}}$, the multiset extension of $\succ_{\mathrm{L}}$.

   Notation: $\succ$ also for $\succ_{\mathrm{L}}$ and $\succ_{\mathrm{C}}$.

The Inference Rules

The resolution calculus $Res^{\succ}_{\mathrm{sel}}$ is parameterized by

- a selection function sel, which is a mapping

$$\mathrm{sel} : C \quad \mapsto \quad \text{set of occurrences of \textit{negative} literals in } C,$$

- and a well-founded ordering $\succ$ on atoms that is total on ground atoms and stable under substitutions.

Ordered Resolution with Selection:

$$\frac{D \vee B \qquad C \vee \neg A}{(D \vee C)\sigma}$$

if the following conditions are satisfied:

(i) $\sigma = \mathrm{mgu}(A, B)$;

(ii) $B\sigma \not\preceq L\sigma$ for all L in D;

(iii) nothing is selected in $D \vee B$ by sel;

(iv) $\neg A$ is selected in $C \vee \neg A$, or nothing is selected in $C \vee \neg A$ and $\neg A\sigma \not\prec L\sigma$ for all L in C.

Ordered Factorization with Selection:

$$\frac{C \vee A \vee B}{(C \vee A)\sigma}$$

if the following conditions are satisfied:

(i) $\sigma = \mathrm{mgu}(A, B)$;

(ii) $A\sigma \not\prec L\sigma$ for all L in C;

(iii) nothing is selected in $C \vee A \vee B$ by sel.

Construction of Candidate Interpretations Let $N, \succ$ be given. We define sets I_C and Δ_C for all ground clauses C over the given signature inductively over $\succ$:

$$I_C \quad := \quad \bigcup_{C \succ D} \Delta_D$$

$$\Delta_C \quad := \quad \begin{cases} \{A\}, & \text{if } C \in N,\ C = C' \vee A,\ A \succ C',\ I_C \not\models C \\ \emptyset, & \text{otherwise} \end{cases}$$

We say that C *produces* A if $\Delta_C = \{A\}$.

The *candidate interpretation* for N (w.r.t. $\succ$) is given as $I^{\succ}_N := \bigcup_C \Delta_C$.

A.8. Redundancy

Let N be a set of ground clauses and C a ground clause (not necessarily in N). C is called *redundant* w.r.t. N if there exist $C_1, \dots, C_n \in N$, $n \geq 0$, such that $C_i \prec C$ and $C_1, \dots, C_n \models C$.

Redundancy for general clauses: C is called *redundant* w.r.t. N if all ground instances $C\sigma$ of C are redundant w.r.t. $G_\Sigma(N)$.

Notation: The set of all clauses that are redundant w.r.t. N is denoted by $\mathit{Red}(N)$.

N is called saturated up to redundancy if the conclusion of every inference from clauses in $N \setminus \mathit{Red}(N)$ is contained in $N \cup \mathit{Red}(N)$.

A.9. Semantic Tableaux

Propositional Expansion Rules

Negation Elimination

$$\frac{\neg\neg F}{F} \qquad \frac{\neg\top}{\bot} \qquad \frac{\neg\bot}{\top}$$

α-Expansion

$$\frac{\alpha}{\begin{array}{c}\alpha_1\\ \alpha_2\end{array}}$$

β-Expansion

$$\frac{\beta}{\beta_1 \mid \beta_2}$$

Classification of Formulas

conjunctive			disjunctive		
α	α_1	α_2	β	β_1	β_2
$F \land G$	F	G	$\neg(F \land G)$	$\neg F$	$\neg G$
$\neg(F \lor G)$	$\neg F$	$\neg G$	$F \lor G$	F	G
$\neg(F \rightarrow G)$	F	$\neg G$	$F \rightarrow G$	$\neg F$	G

We assume that the binary connective $\leftrightarrow$ has been eliminated in advance.

universal		existential	
γ	$\gamma(t)$	δ	$\delta(t)$
$\forall x F$	$F\{x \mapsto t\}$	$\exists x F$	$F\{x \mapsto t\}$
$\neg\exists x F$	$\neg F\{x \mapsto t\}$	$\neg\forall x F$	$\neg F\{x \mapsto t\}$

Expansion Rules Specific to Tableaux with Ground Instantiation

γ-expansion

$$\frac{\gamma}{\gamma(t)} \quad \text{where } t \text{ is some ground term}$$

δ-expansion

$$\frac{\delta}{\delta(c)} \quad \text{where } c \text{ is a new Skolem constant}$$

Expansion Rules Specific to Free-Variable Tableaux

γ-expansion

$$\frac{\gamma}{\gamma(x)} \quad \text{where } x \text{ is a new free variable}$$

δ-expansion

$$\frac{\delta}{\delta(f(x_1,\dots,x_n))}$$

where f is a new Skolem function, and the x_i are the free variables in δ

A.10. E-Algebras

Let E be a set of equations over $\mathrm{T}_\Sigma(X)$. The following inference system allows us to derive consequences of E:

$E \vdash t \approx t$ *(Reflexivity)*
for every $t \in \mathrm{T}_\Sigma(X)$

$$\frac{E \vdash t \approx t'}{E \vdash t' \approx t} \qquad \textit{(Symmetry)}$$

$$\frac{E \vdash t \approx t' \qquad E \vdash t' \approx t''}{E \vdash t \approx t''} \qquad \textit{(Transitivity)}$$

$$\frac{E \vdash t_1 \approx t'_1 \quad \dots \quad E \vdash t_n \approx t'_n}{E \vdash f(t_1,\dots,t_n) \approx f(t'_1,\dots,t'_n)} \qquad \textit{(Congruence)}$$

$E \vdash t\sigma \approx t'\sigma$ *(Instance)*
if $(t \approx t') \in E$ and $\sigma : X \to \mathrm{T}_\Sigma(X)$

A.11. Simplification Orderings

Polynomial Orderings

Instance of the interpretation method:

The carrier set is $U_{\mathcal{A}} = \{n \in \mathbb{N} \mid n \geq 1\}$.

With every function symbol f/n we associate a polynomial $P_f(X_1, \dots, X_n) \in \mathbb{N}[X_1, \dots, X_n]$ with coefficients in $\mathbb{N}$ and indeterminates $X_1, \dots, X_n$. Then we define $f_{\mathcal{A}}(a_1, \dots, a_n) = P_f(a_1, \dots, a_n)$ for $a_i \in U_{\mathcal{A}}$.

If $\operatorname{arity}(f) = 0$, then P_f is a constant ≥ 1.

If $\operatorname{arity}(f) = n \geq 1$, then P_f is a polynomial $P(X_1, \dots, X_n)$, such that every X_i occurs in some monomial $m \cdot X_1^{j_1} \cdots X_k^{j_k}$ with exponent at least 1 and nonzero coefficient $m \in \mathbb{N}$.

The mapping from function symbols to polynomials can be extended to terms: A term t containing the variables $x_1, \dots, x_n$ yields a polynomial P_t with indeterminates $X_1, \dots, X_n$.

Lexicographic Path Ordering Let $\Sigma = (\Omega, \Pi)$ be a finite signature, let $\succ$ be a strict partial ordering *("precedence")* on Ω.

The *lexicographic path ordering* $\succ_{\text{lpo}}$ on $\mathrm{T}_\Sigma(X)$ induced by $\succ$ is defined by: $s \succ_{\text{lpo}} t$ if

(1) $t \in \operatorname{var}(s)$ and $t \neq s$, or

(2) $s = f(s_1, \dots, s_m)$, $t = g(t_1, \dots, t_n)$, and

  (a) $s_i \succeq_{\text{lpo}} t$ for some i, or

  (b) $f \succ g$ and $s \succ_{\text{lpo}} t_j$ for all j, or

  (c) $f = g$, $s \succ_{\text{lpo}} t_j$ for all j, and $(s_1, \dots, s_m) \; (\succ_{\text{lpo}})_{\text{lex}} \; (t_1, \dots, t_n)$.

where $(\succ_{\text{lpo}})_{\text{lex}}$ is the m-fold lexicographic combination of $\succ_{\text{lpo}}$
(note that $f = g$ implies $m = n$).

Knuth–Bendix Ordering Let $\Sigma = (\Omega, \Pi)$ be a finite signature, let $\succ$ be a strict partial ordering *("precedence")* on Ω, let $w : \Omega \cup X \to \mathbb{R}_0^+$ be a *weight function*, such that the following admissibility conditions are satisfied:

$w(x) = w_0 \in \mathbb{R}^+$ for all variables $x \in X$; $w(c) \geq w_0$ for all constants $c \in \Omega$.

If $w(f) = 0$ for some $f/1 \in \Omega$, then $f \succ g$ for all $g/n \in \Omega$ with $f \neq g$.

The weight function w can be extended to terms recursively:

$$w(f(t_1,\dots,t_n)) = w(f) + \sum_{1\le i\le n} w(t_i)$$

or alternatively

$$w(t) = \sum_{x\in\mathrm{var}(t)} w(x)\cdot\#(x,t) + \sum_{f\in\Omega} w(f)\cdot\#(f,t)$$

where $\#(a,t)$ is the number of occurrences of a in t.

The *Knuth–Bendix ordering* $\succ_{\mathrm{kbo}}$ on $\mathrm{T}_\Sigma(X)$ induced by $\succ$ and w is defined by: $s \succ_{\mathrm{kbo}} t$ if

(1) $\#(x,s) \ge \#(x,t)$ for all variables x and $w(s) > w(t)$, or

(2) $\#(x,s) \ge \#(x,t)$ for all variables x, $w(s) = w(t)$, and

  (a) $t = x$, $s = f^n(x)$ for some $n \ge 1$, or

  (b) $s = f(s_1,\dots,s_m)$, $t = g(t_1,\dots,t_n)$, and $f \succ g$, or

  (c) $s = f(s_1,\dots,s_m)$, $t = f(t_1,\dots,t_m)$, and $(s_1,\dots,s_m)\ (\succ_{\mathrm{kbo}})_{\mathrm{lex}}\ (t_1,\dots,t_m)$.

A.12. Dependency Pairs

Given: finite TRS R over $\Sigma = (\Omega,\emptyset)$.

$T_0 := \{t \in \mathrm{T}_\Sigma(X) \mid \exists$ infinite deriv. $t \to_R t_1 \to_R t_2 \to_R \cdots\}$.

$T_\infty := \{t \in T_0 \mid \forall p > \varepsilon : t|_p \notin T_0\}$
$=$ minimal elements of T_0 w.r.t. $\rhd$.

$t \in T_0 \Rightarrow$ there exists a $t' \in T_\infty$ such that $t \trianglerighteq t'$.

$D := \{\mathrm{root}(l) \mid l \to r \in R\}$ is called the set of *defined symbols* of R; $C := \Omega \setminus D$ is called the set of *constructor symbols* of R.

We introduce a new set of function symbols $f^\sharp$ that are only used for the root symbols of this derivation:

$$\Omega^\sharp := \{f^\sharp/n \mid f/n \in \Omega\}.$$

For a term $t = f(t_1,\dots,t_n)$ we define $t^\sharp := f^\sharp(t_1,\dots,t_n)$; for a set of terms T we define $T^\sharp := \{t^\sharp \mid t \in T\}$.

The set of *dependency pairs* of a TRS R is then defined by

$$\mathrm{DP}(R) := \{l^\sharp \to u^\sharp \mid l \to r \in R,\ r \trianglerighteq u,\ u \notin X,\ \mathrm{root}(u) \in D,\ l \not\trianglerighteq u\}.$$

The functions cap and ren are defined by

$$\begin{aligned}
\operatorname{cap}(x) &= x \\
\operatorname{cap}(f(t_1,\dots,t_n)) &= \begin{cases} y & \text{if } f \in D \\ f(\operatorname{cap}(t_1),\dots,\operatorname{cap}(t_n)) & \text{if } f \in C \cup D^\sharp \end{cases}
\end{aligned}$$

$$\begin{aligned}
\operatorname{ren}(x) &= y, \quad y \text{ fresh} \\
\operatorname{ren}(f(t_1,\dots,t_n)) &= f(\operatorname{ren}(t_1),\dots,\operatorname{ren}(t_n))
\end{aligned}$$

A.13. Completion

Critical Pairs Let $l_i \to r_i$ $(i = 1,2)$ be two rewrite rules in a TRS R whose variables have been renamed such that $\operatorname{var}(l_1) \cap \operatorname{var}(l_2) = \emptyset$.

Let $p \in \operatorname{pos}(l_1)$ be a position such that $l_1|_p$ is not a variable and σ is an mgu of $l_1|_p$ and l_2.

Then $r_1\sigma \leftarrow l_1\sigma \to (l_1\sigma)[r_2\sigma]_p$.

$\langle r_1\sigma, (l_1\sigma)[r_2\sigma]_p\rangle$ is called a *critical pair* of R.

$\mathrm{CP}(R)$ denotes the set of all critical pairs between rules in R.

Knuth–Bendix Completion The completion procedure is presented as a set of inference rules working on a set of equations E and a set of rules R: $E_0, R_0 \vdash E_1, R_1 \vdash E_2, R_2 \vdash \dots$

At the beginning, $E = E_0$ is the input set and $R = R_0$ is empty. At the end, E should be empty; then R is the result.

For each step $E, R \vdash E', R'$, the equational theories of $E \cup R$ and $E' \cup R'$ agree: $\approx_{E \cup R} = \approx_{E' \cup R'}$.

Notation: The formula $s \mathrel{\dot{\approx}} t$ denotes either $s \approx t$ or $t \approx s$.

Orient:

$$\frac{E \cup \{s \mathrel{\dot{\approx}} t\}, \quad R}{E, \quad R \cup \{s \to t\}} \qquad \text{if } s \succ t$$

Delete:

$$\frac{E \cup \{s \approx s\}, \quad R}{E, \quad R}$$

Deduce:

$$\frac{E, \quad R}{E \cup \{s \approx t\}, \quad R} \qquad \text{if } \langle s, t\rangle \in \mathrm{CP}(R).$$

Simplify-Eq:

$$\frac{E \cup \{s \dot{\approx} t\}, \quad R}{E \cup \{u \approx t\}, \quad R} \qquad \text{if } s \to_R u.$$

R-Simplify-Rule:

$$\frac{E, \quad R \cup \{s \to t\}}{E, \quad R \cup \{s \to u\}} \qquad \text{if } t \to_R u.$$

L-Simplify-Rule:

$$\frac{E, \quad R \cup \{s \to t\}}{E \cup \{u \approx t\}, \quad R} \qquad \begin{array}{l}\text{if } s \to_R u \text{ using a rule } l \to r \in R \\ \text{such that } s \sqsupset l.\end{array}$$

The *encompassment quasi-ordering* $\underset{\sim}{\sqsupset}$ is defined by

$$s \underset{\sim}{\sqsupset} l \ \text{ if } \ s|_p = l\sigma \text{ for some } p \text{ and } \sigma$$

and $\sqsupset = \underset{\sim}{\sqsupset} \setminus \underset{\sim}{\sqsubset}$ is the strict part of $\underset{\sim}{\sqsupset}$.

Semicritical Pairs Let $u_i \dot{\approx} v_i$ $(i = 1, 2)$ be equations in E whose variables have been renamed such that $\text{var}(u_1 \dot{\approx} v_1) \cap \text{var}(u_2 \dot{\approx} v_2) = \emptyset$. Let $p \in \text{pos}(u_1)$ be a position such that $u_1|_p$ is not a variable, σ is an mgu of $u_1|_p$ and u_2, and $u_i\sigma \not\preceq v_i\sigma$ $(i = 1, 2)$. Then $\langle v_1\sigma, (u_1\sigma)[v_2\sigma]_p\rangle$ is called a *semicritical pair* of E with respect to $\succ$.

The set of all semicritical pairs of E is denoted by $\text{SP}_\succ(E)$.

Unfailing Completion The "Deduce" rule now takes the following form:

Deduce:

$$\frac{E, \quad R}{E \cup \{s \approx t\}, \quad R} \qquad \text{if } \langle s, t\rangle \in \text{SP}_\succ(E \cup R).$$

A.14. Superposition

The Inferene Rules

Pos. Superposition:
$$\frac{D' \vee t \approx t' \qquad C' \vee s[u] \approx s'}{(D' \vee C' \vee s[t'] \approx s')\sigma}$$
where $\sigma = \mathrm{mgu}(t, u)$ and u is not a variable.

Neg. Superposition:
$$\frac{D' \vee t \approx t' \qquad C' \vee s[u] \not\approx s'}{(D' \vee C' \vee s[t'] \not\approx s')\sigma}$$
where $\sigma = \mathrm{mgu}(t, u)$ and u is not a variable.

Equality Resolution:
$$\frac{C' \vee s \not\approx s'}{C'\sigma}$$
where $\sigma = \mathrm{mgu}(s, s')$.

Equality Factoring:
$$\frac{C' \vee s' \approx t' \vee s \approx t}{(C' \vee t \not\approx t' \vee s \approx t')\sigma}$$
where $\sigma = \mathrm{mgu}(s, s')$.

Clause Orderings

Let $\succ$ be a *reduction ordering that is total on ground terms.*

To a positive literal $s \approx t$, we assign the multiset $\{s, t\}$, to a negative literal $s \not\approx t$ the multiset $\{s, s, t, t\}$. The *literal ordering* $\succ_{\mathrm{L}}$ compares these multisets using the multiset extension of $\succ$.

The *clause ordering* $\succ_{\mathrm{C}}$ compares clauses by comparing their multisets of literals using the multiset extension of $\succ_{\mathrm{L}}$.

The Ordering Restrictions Inferences have to be computed only if the following ordering restrictions are satisfied (after applying the unifier to the premises):

– In superposition inferences, the left premise is not greater than or equal to the right one.

– The last literal in each premise is maximal in the respective premise, i.e., there exists no greater literal (strictly maximal for positive literals in superposition inferences, i.e., there exists no greater or equal literal).

– In these literals, the lhs is neither smaller than nor equal to the rhs (except in equality resolution inferences).

Construction of Candidate Interpretations Let N be a set of clauses not containing $\bot$. Using induction on the clause ordering we define sets of rewrite rules E_C and R_C for all $C \in G_\Sigma(N)$ as follows:

Assume that E_D has already been defined for all $D \in G_\Sigma(N)$ with $D \prec_{\mathrm{C}} C$. Then $R_C = \bigcup_{D \prec_{\mathrm{C}} C} E_D$.

The set E_C contains the rewrite rule $s \to t$ if

(a) $C = C' \vee s \approx t$.

(b) $s \approx t$ is strictly maximal in C.

(c) $s \succ t$.

(d) C is false in R_C.

(e) C' is false in $R_C \cup \{s \to t\}$.

(f) s is irreducible w.r.t. R_C.

In this case, C is called *productive*. Otherwise $E_C = \emptyset$.

Finally, $R_\infty = \bigcup_{D \in G_\Sigma(N)} E_D$.