3.8 Inference Systems and Proofs

Inference systems Γ (proof calculi) are sets of tuples

$$(F_1, \ldots, F_n, F_{n+1}),\ n \geq 0,$$

called *inferences*, and written

$$\frac{\overbrace{F_1 \ \ldots \ F_n}^{\text{premises}}}{\underbrace{F_{n+1}}_{\text{conclusion}}}\ .$$

Clausal inference system: premises and conclusions are clauses. One also considers inference systems over other data structures.

Inference Systems

Inference systems Γ are shorthands for reduction systems over sets of formulas. If N is a set of formulas, then

$$\frac{\overbrace{F_1 \ \ldots \ F_n}^{\text{premises}}}{\underbrace{F_{n+1}}_{\text{conclusion}}}\quad \textit{side condition}$$

is a shorthand for

$$N \cup \{F_1, \ldots, F_n\} \quad \Rightarrow_\Gamma \quad N \cup \{F_1, \ldots, F_n\} \cup \{F_{n+1}\}$$
$$\text{if } \textit{side condition}$$

Proofs

A *proof* in Γ of a formula F from a set of formulas N (called *assumptions*) is a sequence $F_1, \ldots, F_k$ of formulas where

(i) $F_k = F$,

(ii) for all $1 \leq i \leq k$: $F_i \in N$ or there exists an inference

$$\frac{F_{m_1} \ \ldots \ F_{m_n}}{F_i}$$

in Γ, such that $0 \leq m_j < i$, for $1 \leq j \leq n$.

Soundness and Completeness

Provability $\vdash_\Gamma$ of F from N in Γ:

$N \vdash_\Gamma F$ if there exists a proof Γ of F from N.

Γ is called *sound*, if

$$\frac{F_1 \ \ldots \ F_n}{F} \in \Gamma \text{ implies } F_1, \ldots, F_n \models F$$

Γ is called *complete*, if

$N \models F$ implies $N \vdash_\Gamma F$

Γ is called *refutationally complete*, if

$N \models \bot$ implies $N \vdash_\Gamma \bot$

Proposition 3.12

(i) Let Γ be sound. Then $N \vdash_\Gamma F \Rightarrow N \models F$

(ii) If $N \vdash_\Gamma F$ then there exist finitely many $F_1, \ldots, F_n \in N$ such that $F_1, \ldots, F_n \vdash_\Gamma F$

Reduced Proofs

The definition of a proof of F given above admits sequences $F_1, \ldots, F_k$ of formulas where some F_i are not ancestors of $F_k = F$ (i. e., some F_i are not actually used to derive F).

A proof is called *reduced*, if every F_i with $i < k$ is an ancestor of F_k.

We obtain a reduced proof from a proof by marking first F_k and then recursively all the premises used to derive a marked conclusion, and by deleting all non-marked formulas in the end.

Reduced Proofs as Trees

$$\begin{array}{rcl} \text{markings} & \widehat{=} & \text{formulas} \\ \text{leaves} & \widehat{=} & \text{assumptions and axioms} \\ \text{other nodes} & \widehat{=} & \text{inferences: conclusion } \widehat{=} \text{ parent node} \\ & & \text{premises } \widehat{=} \text{ child nodes} \end{array}$$

$$\dfrac{\dfrac{P(f(c)) \lor Q(b) \quad \neg P(f(c)) \lor \neg P(f(c)) \lor Q(b)}{\neg P(f(c)) \lor Q(b) \lor Q(b)}}{\dfrac{P(f(c)) \lor Q(b) \qquad \neg P(f(c)) \lor Q(b)}{\dfrac{Q(b) \lor Q(b)}{\dfrac{Q(b) \qquad \neg P(f(c)) \lor \neg Q(b)}{\dfrac{P(f(c)) \qquad \neg P(f(c))}{\bot}}}}}$$

3.9 Ground (or propositional) Resolution

We observe that propositional clauses and ground clauses are essentially the same, as long as we do not consider equational atoms.

In this section we only deal with ground clauses.

Note that unlike in Section 2 we admit duplicated literals in clauses, i. e., we treat clauses like multisets of literals, not like sets.

The Resolution Calculus Res

Resolution inference rule:

$$\dfrac{D \lor A \qquad C \lor \neg A}{D \lor C}$$

Terminology: $D \lor C$: *resolvent*; A: *resolved atom*

(Positive) factorization inference rule:

$$\dfrac{C \lor A \lor A}{C \lor A}$$

These are *schematic inference rules*; for each substitution of the *schematic variables* C, D, and A, by ground clauses and ground atoms, respectively, we obtain an inference.

We treat "$\lor$" as associative and commutative, hence A and $\neg A$ can occur anywhere in the clauses; moreover, when we write $C \lor A$, etc., this includes unit clauses, that is, $C = \bot$.

Sample Refutation

1.	$\neg P(f(c)) \vee \neg P(f(c)) \vee Q(b)$	(given)
2.	$P(f(c)) \vee Q(b)$	(given)
3.	$\neg P(g(b,c)) \vee \neg Q(b)$	(given)
4.	$P(g(b,c))$	(given)
5.	$\neg P(f(c)) \vee Q(b) \vee Q(b)$	(Res. 2. into 1.)
6.	$\neg P(f(c)) \vee Q(b)$	(Fact. 5.)
7.	$Q(b) \vee Q(b)$	(Res. 2. into 6.)
8.	$Q(b)$	(Fact. 7.)
9.	$\neg P(g(b,c))$	(Res. 8. into 3.)
10.	$\bot$	(Res. 4. into 9.)

Soundness of Resolution

Theorem 3.13 *Propositional resolution is sound.*

Proof. Let $\mathcal{B} \in \Sigma$-Alg. To be shown:

(i) for resolution: $\mathcal{B} \models D \vee A,\ \mathcal{B} \models C \vee \neg A \ \Rightarrow\ \mathcal{B} \models D \vee C$

(ii) for factorization: $\mathcal{B} \models C \vee A \vee A \ \Rightarrow\ \mathcal{B} \models C \vee A$

(i): Assume premises are valid in $\mathcal{B}$. Two cases need to be considered:
If $\mathcal{B} \models A$, then $\mathcal{B} \models C$, hence $\mathcal{B} \models D \vee C$.
Otherwise, $\mathcal{B} \models \neg A$, then $\mathcal{B} \models D$, and again $\mathcal{B} \models D \vee C$.
(ii): even simpler. □

Note: In propositional logic (ground clauses) we have:

1. $\mathcal{B} \models L_1 \vee \ldots \vee L_n$ if and only if there exists i: $\mathcal{B} \models L_i$.
2. $\mathcal{B} \models A$ or $\mathcal{B} \models \neg A$.

This does not hold for formulas with variables!

3.10 Refutational Completeness of Resolution

How to show refutational completeness of propositional resolution:

- We have to show: $N \models \bot \ \Rightarrow\ N \vdash_{Res} \bot$, or equivalently: If $N \not\vdash_{Res} \bot$, then N has a model.
- Idea: Suppose that we have computed sufficiently many inferences (and not derived $\bot$).

- Now order the clauses in N according to some appropriate ordering, inspect the clauses in ascending order, and construct a series of Herbrand interpretations.
- The limit interpretation can be shown to be a model of N.

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 lenght-based ordering on atoms when these are viewed as words over a suitable alphabet.)
2. Extend $\succ$ to an *ordering* $\succ_L$ *on ground literals:*

$$\begin{array}{lll} [\neg]A & \succ_L & [\neg]B \quad \text{, if } A \succ B \\ \neg A & \succ_L & A \end{array}$$

3. Extend $\succ_L$ to an *ordering* $\succ_C$ *on ground clauses:*
   $\succ_C = (\succ_L)_{\text{mul}}$, the multiset extension of $\succ_L$.

   Notation: $\succ$ also for $\succ_L$ and $\succ_C$.

Example

Suppose $A_5 \succ A_4 \succ A_3 \succ A_2 \succ A_1 \succ A_0$. Then:

$$\begin{array}{ll} & A_5 \vee \neg A_5 \\ \succ & A_3 \vee \neg A_4 \\ \succ & \neg A_1 \vee A_3 \vee A_4 \\ \succ & \neg A_1 \vee A_2 \\ \succ & A_1 \vee A_2 \\ \succ & A_0 \vee A_1 \end{array}$$

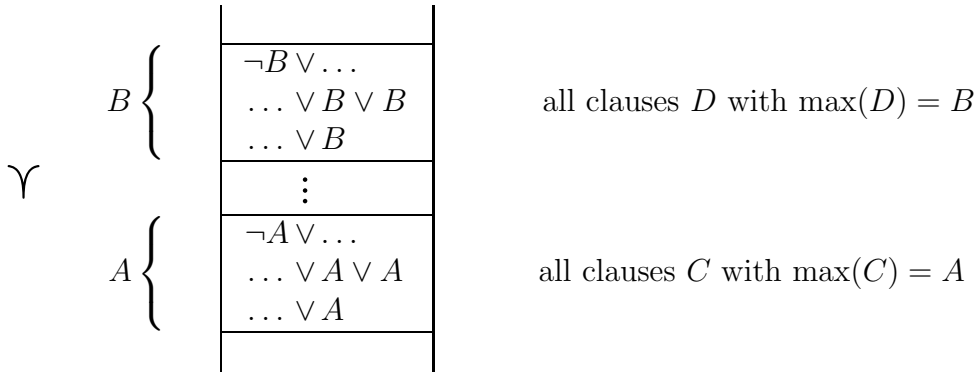
Properties of the Clause Ordering

Proposition 3.14

1. *The orderings on literals and clauses are total and well-founded.*
2. *Let C and D be clauses with $A = \max(C)$, $B = \max(D)$, where $\max(C)$ denotes the maximal atom in C.*
   (i) If $A \succ B$ then $C \succ D$.
   (ii) If $A = B$, A occurs negatively in C but only positively in D, then $C \succ D$.

Stratified Structure of Clause Sets

Let $B \succ A$. Clause sets are then stratified in this form:



Closure of Clause Sets under Res

$$
\begin{aligned}
Res(N) &= \{\, C \mid C \text{ is conclusion of an inference in } Res \\
&\qquad\qquad \text{with premises in } N \,\} \\
Res^0(N) &= N \\
Res^{n+1}(N) &= Res(Res^n(N)) \cup Res^n(N), \text{ for } n \geq 0 \\
Res^*(N) &= \textstyle\bigcup_{n\geq 0} Res^n(N)
\end{aligned}
$$

N is called *saturated* (w. r. t. resolution), if $Res(N) \subseteq N$.

Proposition 3.15

(i) $Res^*(N)$ *is saturated.*

(ii) *Res is refutationally complete, iff for each set* N *of ground clauses:*

$$N \models \bot \text{ iff } \bot \in Res^*(N)$$

Construction of Interpretations

Given: set N of ground clauses, atom ordering $\succ$.
Wanted: Herbrand interpretation I such that

- "many" clauses from N are valid in I;
- $I \models N$, *if* N *is saturated and* $\bot \notin N$.

Construction according to $\succ$, starting with the smallest clause.

Main Ideas of the Construction

- Clauses are considered in the order given by $\succ$.
- When considering C, one already has a partial interpretation I_C (initially $I_C = \emptyset$) available.
- If C is true in the partial interpretation I_C, nothing is done. ($\Delta_C = \emptyset$).
- If C is false, one would like to change I_C such that C becomes true.
- Changes should, however, be *monotone*. One never deletes anything from I_C and the truth value of clauses smaller than C should be maintained the way it was in I_C.
- Hence, one chooses $\Delta_C = \{A\}$ if, and only if, C is false in I_C, if A occurs positively in C (*adding A will make C become true*) and if this occurrence in C is strictly maximal in the ordering on literals (*changing the truth value of A has no effect on smaller clauses*).
- We say that the construction fails for a clause C, if C is false in I_C and $\Delta_C = \emptyset$.
- We will show: If there are clauses for which the construction fails, then some inference with the smallest such clause (the so-called "minimal counterexample") has not been computed. Otherwise, the limit interpretation is a model of all clauses.

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 $\succ$:

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

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

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

The *candidate interpretation* for N (w. r. t. $\succ$) is given as $I_N^{\succ} := \bigcup_C \Delta_C$. (We also simply write I_N or I for $I_N^{\succ}$ if $\succ$ is either irrelevant or known from the context.)

Example

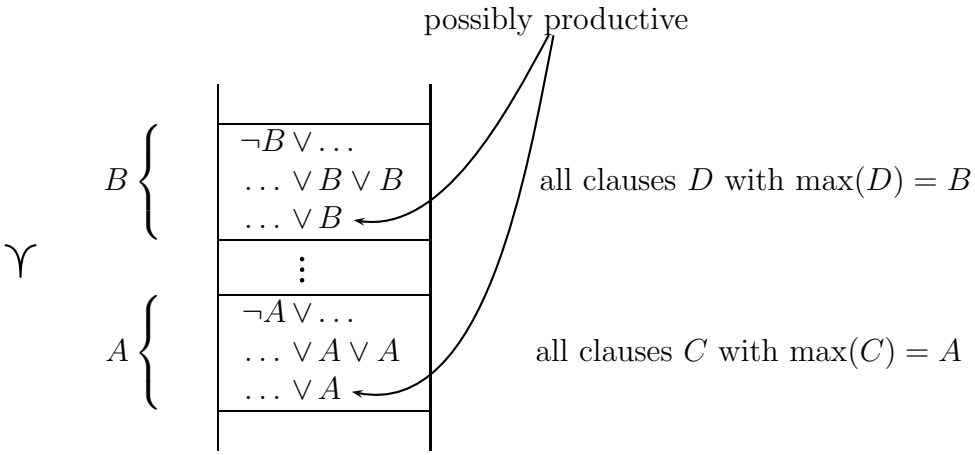
Let $A_5 \succ A_4 \succ A_3 \succ A_2 \succ A_1 \succ A_0$ (max. literals in red)

	clauses C	I_C	Δ_C	Remarks
7	$\neg A_1 \vee A_5$	$\{A_1, A_2, A_4\}$	$\{A_5\}$	
6	$\neg A_1 \vee A_3 \vee \neg A_4$	$\{A_1, A_2, A_4\}$	$\emptyset$	max. lit. $\neg A_4$ neg.; *min. counter-ex.*
5	$A_0 \vee \neg A_1 \vee A_3 \vee A_4$	$\{A_1, A_2\}$	$\{A_4\}$	A_4 maximal
4	$\neg A_1 \vee A_2$	$\{A_1\}$	$\{A_2\}$	A_2 maximal
3	$A_1 \vee A_2$	$\{A_1\}$	$\emptyset$	true in I_C
2	$A_0 \vee A_1$	$\emptyset$	$\{A_1\}$	A_1 maximal
1	$\neg A_0$	$\emptyset$	$\emptyset$	true in I_C

$I = \{A_1, A_2, A_4, A_5\}$ is not a model of the clause set
$\Rightarrow$ there exists a *counterexample*.

Structure of $N, \succ$

Let $B \succ A$; producing a new atom does not affect smaller clauses.



Some Properties of the Construction

Proposition 3.16

(i) If $C = C' \vee \neg A$, then no $D \succeq C$ produces A.

(ii) If C is productive, then $I_C \cup \Delta_C \models C$.

(iii) *Let* $D' \succeq D \succeq C$. *Then*

$$I_D \cup \Delta_D \models C \text{ implies } I_{D'} \cup \Delta_{D'} \models C \text{ and } I_N \models C.$$

If, in addition, $C \in N$ *or* $\max(D) \succ \max(C)$, *then*

$$I_D \cup \Delta_D \not\models C \text{ implies } I_{D'} \cup \Delta_{D'} \not\models C \text{ and } I_N \not\models C.$$

(iv) *Let* $D' \succeq D \succ C$. *Then*

$$I_D \models C \text{ implies } I_{D'} \models C \text{ and } I_N \models C.$$

If, in addition, $C \in N$ *or* $\max(D) \succ \max(C)$, *then*

$$I_D \not\models C \text{ implies } I_{D'} \not\models C \text{ and } I_N \not\models C.$$

(v) *If* $D = D' \vee A$ *produces* A, *then* $I_C \not\models D'$ *for every* $C \succ D$ *and* $I_N \not\models D'$.

(vi) *If for every clause* $C \in N$, C *is productive or* $I_C \models C$, *then* $I_N \models N$.

Model Existence Theorem

Theorem 3.17 (Bachmair & Ganzinger 1990) *Let* $\succ$ *be a clause ordering, let* N *be saturated w. r. t. Res, and suppose that* $\bot \notin N$. *Then* $I_N^{\succ} \models N$.

Proof. Suppose $\bot \notin N$, but $I_N^{\succ} \not\models N$. Let $C \in N$ minimal (w. r. t. $\succ$) such that C is neither productive nor $I_C \models C$. As $C \neq \bot$ there exists a maximal literal in C. There are two possible reasons why C is not productive:

Case 1: The maximal literal $\neg A$ is negative, i. e., $C = C' \vee \neg A$. Then $I_C \models A$ and $I_C \not\models C'$. So some $D = D' \vee A \in N$ with $C \succ D$ produces A, and $I_C \not\models D'$. The inference

$$\frac{D' \vee A \qquad C' \vee \neg A}{D' \vee C'}$$

yields a clause $D' \vee C' \in N$ that is smaller than C. As $I_C \not\models D' \vee C'$, we know that $D' \vee C'$ is neither productive nor $I_{D' \vee C'} \models D' \vee C'$. This contradicts the minimality of C.

Case 2: The maximal literal A is positive, but not strictly maximal, i. e., $C = C' \vee A \vee A$. Then there is an inference

$$\frac{C' \vee A \vee A}{C' \vee A}$$

that yields a smaller clause $C' \vee A \in N$. As $I_C \not\models C' \vee A$, this clause is neither productive nor $I_{C' \vee A} \models C' \vee A$. Since $C \succ C' \vee A$, this contradicts the minimality of C. □

Corollary 3.18 *Let* N *be saturated w. r. t. Res. Then* $N \models \bot$ *if and only if* $\bot \in N$.

Compactness of Propositional Logic

Lemma 3.19 *Let N be a set of propositional (or first-order ground) clauses. Then N is unsatisfiable, if and only if some finite subset $N' \subseteq N$ is unsatisfiable.*

Proof. The "if" part is trivial. For the "only if" part, assume that N be unsatisfiable. Consequently, $Res^*(N)$ unsatisfiable as well. By refutational completeness of resolution, $\bot \in Res^*(N)$. So there exists an $n \geq 0$ such that $\bot \in Res^n(N)$, which means that $\bot$ has a finite resolution proof. Now choose N' as the set of assumptions in this proof. □

Theorem 3.20 (Compactness for Propositional Formulas) *Let S be a set of propositional (or first-order ground) formulas. Then S is unsatisfiable, if and only if some finite subset $S' \subseteq S$ is unsatisfiable.*

Proof. The "if" part is again trivial. For the "only if" part, assume that S be unsatisfiable. Transform S into an equivalent set N of clauses. By the previous lemma, N has a finite unsatisfiable subset N'. Now choose for every clause C in N' one formula F of S such that C is contained in the CNF of F. Let S' be the set of these formulas. □

3.11 General Resolution

Propositional (ground) resolution:

- refutationally complete,
- in its most naive version: not guaranteed to terminate for satisfiable sets of clauses, (improved versions do terminate, however)
- inferior to the DPLL procedure.

But: in contrast to the DPLL procedure, resolution can be easily extended to non-ground clauses.

Two Lemmas

Lemma 3.21 *Let $\mathcal{A}$ be a Σ-algebra and let F be a Σ-formula with free variables $x_1, \ldots, x_n$. Then*

$$\mathcal{A} \models \forall x_1, \ldots, x_n F \quad \text{if and only if} \quad \mathcal{A} \models F$$

Lemma 3.22 *Let F be a Σ-formula with free variables $x_1, \ldots, x_n$, let σ be a substitution, and let $y_1, \ldots, y_m$ be the free variables of $F\sigma$. Then*

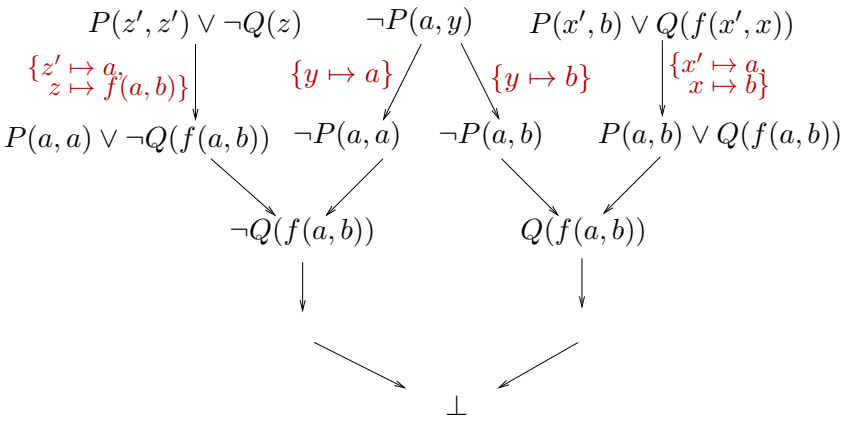
$$\mathcal{A} \models \forall x_1, \ldots, x_n F \quad \text{implies} \quad \mathcal{A} \models \forall y_1, \ldots, y_m F\sigma$$

In particular, if $\mathcal{A}$ is a model of an (implicitly universally quantified) clause C, then it is also a model of all (implicitly universally quantified) instances $C\sigma$ of C.

Consequently, if we show that some instances of clauses in a set N are unsatisfiable, then we have also shown that N itself is unsatisfiable.

General Resolution through Instantiation

Idea: instantiate clauses appropriately:



Problems:

More than one instance of a clause can participate in a proof.

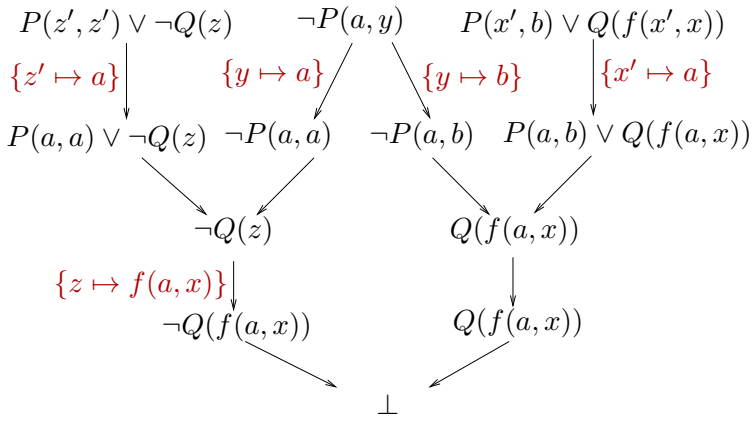
Even worse: There are infinitely many possible instances.

Observation:

Instantiation must produce complementary literals (so that inferences become possible).

Idea:

Do not instantiate more than necessary to get complementary literals.



Lifting Principle

Problem: Make saturation of infinite sets of clauses as they arise from taking the (ground) instances of finitely many *general* clauses (with variables) effective and efficient.

Idea (Robinson 1965):

- Resolution for general clauses:
- *Equality* of ground atoms is generalized to *unifiability* of general atoms;
- Only compute *most general* (minimal) unifiers (mgu).

Significance: The advantage of the method in (Robinson 1965) compared with (Gilmore 1960) is that unification enumerates only those instances of clauses that participate in an inference. Moreover, clauses are not right away instantiated into ground clauses. Rather they are instantiated only as far as required for an inference. Inferences with non-ground clauses in general represent infinite sets of ground inferences which are computed simultaneously in a single step.

Unification

Let $E = \{s_1 \doteq t_1, \ldots, s_n \doteq t_n\}$ (s_i, t_i terms or atoms) a multiset of *equality problems*. A substitution σ is called a *unifier* of E if $s_i\sigma = t_i\sigma$ for all $1 \leq i \leq n$.

If a unifier of E exists, then E is called *unifiable*.

A substitution σ is called *more general* than a substitution τ, denoted by $\sigma \leq \tau$, if there exists a substitution ρ such that $\rho \circ \sigma = \tau$, where $(\rho \circ \sigma)(x) := (x\sigma)\rho$ is the composition of σ and ρ as mappings. (Note that $\rho \circ \sigma$ has a finite domain as required for a substitution.)

If a unifier of E is more general than any other unifier of E, then we speak of a *most general unifier* of E, denoted by $\mathrm{mgu}(E)$.

Proposition 3.23

(i) $\leq$ *is a quasi-ordering on substitutions, and* $\circ$ *is associative.*

(ii) If $\sigma \leq \tau$ *and* $\tau \leq \sigma$ *(we write* $\sigma \sim \tau$ *in this case), then* $x\sigma$ *and* $x\tau$ *are equal up to (bijective) variable renaming, for any* x *in* X*.*

A substitution σ is called *idempotent*, if $\sigma \circ \sigma = \sigma$.

Proposition 3.24 *σ is idempotent iff $dom(\sigma) \cap codom(\sigma) = \emptyset$.*

Rule-Based Naive Standard Unification

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

SU: Main Properties

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

Proposition 3.25 *If E is a solved form then σ_E is an mgu of E.*

Theorem 3.26

1. *If $E \Rightarrow_{SU} E'$ then σ is a unifier of E iff σ is a unifier of E'*
2. *If $E \Rightarrow^*_{SU} \bot$ then E is not unifiable.*
3. *If $E \Rightarrow^*_{SU} E'$ with E' in solved form, then $\sigma_{E'}$ is an mgu of E.*

Proof. (1) We have to show this for each of the rules. Let's treat the case for the 4th rule here. Suppose σ is a unifier of $x \doteq t$, that is, $x\sigma = t\sigma$. Thus, $\sigma \circ \{x \mapsto t\} = \sigma[x \mapsto t\sigma] = \sigma[x \mapsto x\sigma] = \sigma$. Therefore, for any equation $u \doteq v$ in E: $u\sigma = v\sigma$, iff $u\{x \mapsto t\}\sigma = v\{x \mapsto t\}\sigma$. (2) and (3) follow by induction from (1) using Proposition 3.25. □

Main Unification Theorem

Theorem 3.27 *E is unifiable if and only if there is a most general unifier σ of E, such that σ is idempotent and $dom(\sigma) \cup codom(\sigma) \subseteq var(E)$.*

Proof.

- $\Rightarrow_{SU}$ is terminating. A suitable lexicographic ordering on the multisets E (with $\bot$ minimal) shows this. Compare in this order:
  (1) the number of variables that occur in E below a function or predicate symbol, or on the right-hand side of an equation, or at least twice;
  (2) the multiset of the sizes (numbers of symbols) of all equations in E;
  (3) the number of non-variable left-hand sides of equations in E.
- A system E that is irreducible w. r. t. $\Rightarrow_{SU}$ is either $\bot$ or a solved form.
- Therefore, reducing any E by SU will end (no matter what reduction strategy we apply) in an irreducible E' having the same unifiers as E, and we can read off the mgu (or non-unifiability) of E from E' (Theorem 3.26, Proposition 3.25).
- σ is idempotent because of the substitution in rule 4. $dom(\sigma) \cup codom(\sigma) \subseteq var(E)$, as no new variables are generated.

□

Rule-Based Polynomial Unification

Problem: using $\Rightarrow_{SU}$, an *exponential growth* of terms is possible.

The following unification algorithm avoids this problem, at least if the final solved form is represented as a DAG.

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

Properties of PU

Theorem 3.28

1. *If $E \Rightarrow_{PU} E'$ then σ is a unifier of E iff σ is a unifier of E'*
2. *If $E \Rightarrow_{PU}^* \bot$ then E is not unifiable.*
3. *If $E \Rightarrow_{PU}^* E'$ with E' in solved form, then $\sigma_{E'}$ is an mgu of E.*

Note: The solved form of $\Rightarrow_{PU}$ is different from the solved form obtained from $\Rightarrow_{SU}$. In order 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 $\{x_1 \mapsto t_1\} \circ \cdots \circ \{x_k \mapsto t_k\}$.

Resolution for General Clauses

We obtain the resolution inference rules for non-ground clauses from the inference rules for ground clauses by replacing equality by unifiabilty:

General binary resolution Res:

$$\frac{D \vee B \qquad C \vee \neg A}{(D \vee C)\sigma} \quad \text{if } \sigma = \mathrm{mgu}(A, B) \qquad \text{[resolution]}$$

$$\frac{C \vee A \vee B}{(C \vee A)\sigma} \qquad \text{if } \sigma = \mathrm{mgu}(A, B) \quad \text{[factorization]}$$

For inferences with more than one premise, we assume that the variables in the premises are (bijectively) renamed such that they become different to any variable in the other premises. We do not formalize this. Which names one uses for variables is otherwise irrelevant.

Lifting Lemma

Lemma 3.29 *Let C and D be variable-disjoint clauses. If*

$$\frac{\begin{array}{cc} D & C \\ \downarrow \sigma & \downarrow \rho \\ D\sigma & C\rho \end{array}}{C'} \qquad \textit{[propositional resolution]}$$

then there exists a substitution τ such that

$$\frac{\begin{array}{cc} D & C \end{array}}{\begin{array}{c} C'' \\ \downarrow \tau \\ C' = C''\tau \end{array}} \qquad \textit{[general resolution]}$$

An analogous lifting lemma holds for factorization.