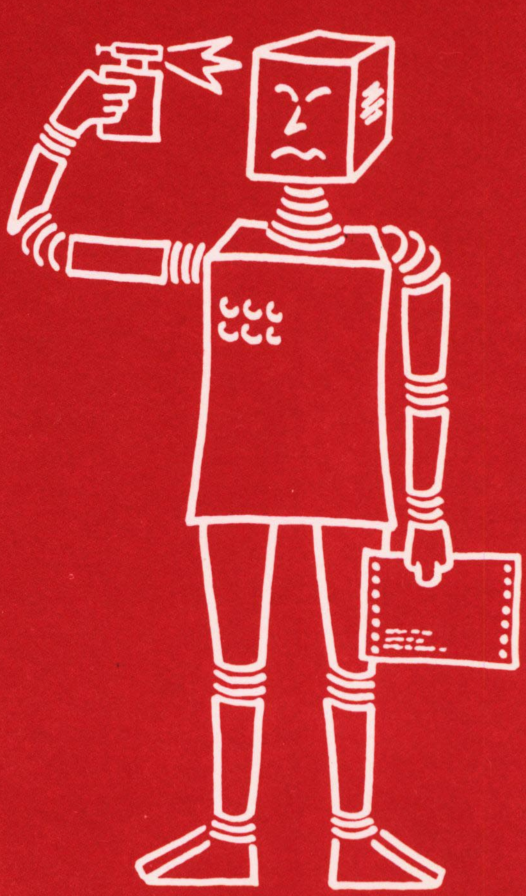
SEKI-PROJEKT

SEKI MEMO

Universität Kaiserslautern
Fachbereich Informatik
Postfach 3049
D-6750 Kaiserslautern
West Germany

Universität Karlsruhe
Institut für Informatik I
Postfach 6380
D-7500 Karlsruhe
West Germany



THEORY UNIFICATION IN ABSTRACT CLAUSE GRAPHS

Hans Jürgen Ohlbach

MEMO SEKI - 85 - I - KL

Theory Unification in Abstract Clause Graphs

Interner Bericht

Hans Jürgen Ohlbach
FB Informatik
University of Kaiserslautern
675 Kaiserslautern
West Germany

Abstract

Clause Graphs, as they were defined in the 1970s, are graphs representing first order formulas in conjunctive normal form together with the resolution possibilities. The nodes are labelled with literals and the edges (links) connect complementary unifiable literals. This report describes a generalization of this concept, called abstract clause graphs. The nodes of abstract clause graphs are still labelled with literals, the links however connect literals that are "unifiable" relative to a given relation between literals. This relation is not explicitely defined; only certain "abstract" properties are required, for instance the existence of a special purpose unification algorithm is assumed which computes substitutions, the application of which makes the relation hold for two literals.
When instances of already existing literals are added to the graph (e.g. due to resolution or factoring), the links to the new literals are derived from the links of their ancestors. An inheritance mechanism for such links is presented which operates only on the attached substitutions and does not have to unify the literals. This solves a long standing open problem of connection graph calculi: how to inherit links (with several unifiers attached) such that no unifier has to be computed more than once.

Contents

1. Introduction and Motivation

A connection graph in the sense of Robert Kowalski [Ko75] consists of nodes labelled with literals (as constituents of clauses) and links, which connect complementary unifiable literals in different clauses.

Since the literals have to be Robinson unifiable, there is (up to the renaming of variables) only one most general unifier (mgu) and it is assumed that this single unifier is attached to the link.

A typical operation upon such a graph is resolution: A link is selected, the resolvent is generated, and the new links are created by inheritance from the links connected to the parent clauses. Finally the link resolved upon with its mgu is removed from the graph. The inheritance mechanism avoids searching the whole graph for potentially complementary literals; the new unifiers still have to be calculated by literal unification, however. In 1975 Bruynooghe presented a mechanism for inheriting Robinson unifiers directly without literal unification [Br75]. But in the last few years different extensions of the connection graph calculus have been proposed that make it necessary to reformulate the inheritance mechanism.

The main extensions are:

1. Other link types were introduced, connecting not only complementary unifiable literals (i.e. the atoms are unifiable and the signs are different) in different clauses, but also directly unifiable literals (i.e. the atoms are unifiable and the signs are equal) in different clauses as well as in the same clause etc. [Ei81], [Wa82]. For every new link type an inheritance mechanism of its own had to be defined.

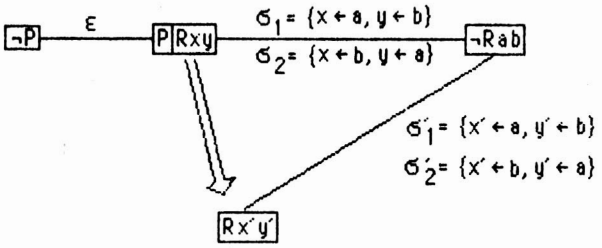
2. The unification concept was augmented. New unification algorithms are able to exploit certain properties of functions and predicates [Si84]. For instance, terms like f(a,f(b,c)) and f(f(b,c),a) are unifiable, provided f is associative and commutative. But also literals like a<b and a>b are complementary unifiable under the usual interpretation of the < and > predicates [St83].

3. The underlying logical calculus has been augmented by sorts [Wa82a] and polymorphic functions [SS85].

A very unpleasant consequence of these extensions is the fact that there is not only one most general unifier, but possibly an infinite number of independent ones. Links in the extended clause graph calculus are therefore labelled with a set of mgus. After resolving with one of these mgus, it is no longer possible to remove the whole link from the graph, but only this single mgu. This leads to the undesired effect that ***a unifier once removed may suddenly reappear in an inherited link***, as the following example demonstrates:

1.1 Example (Reappearance of a removed unifier in an inherited link)

Let R be a symmetric predicate.



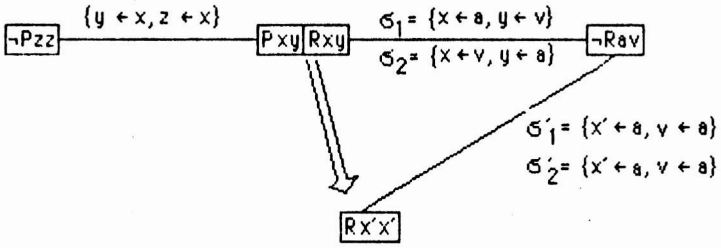
¬P
ε
P
Rxy
$\sigma_1 = \{x \leftarrow a, y \leftarrow b\}$
$\sigma_2 = \{x \leftarrow b, y \leftarrow a\}$
¬Rab
$\sigma_1' = \{x' \leftarrow a, y' \leftarrow b\}$
$\sigma_2' = \{x' \leftarrow b, y' \leftarrow a\}$
Rx'y'

Unification of Rx'y' and ¬Rab after resolution upon the predicate P generates σ_1' and σ_2' regardless whether σ_1 or σ_2 has been previously removed or not. ■

In order to prevent this effect, which introduces the kind of redundancy again the connection graph procedure was trying to eliminate in the first place, it is necessary to inherit the unifiers individually such that a direct relation between the parent unifier and its descendants can be established. If this is possible we can not only prevent the reappearance of removed mgus, but further deletion rules will become possible.

1.2 Example (for a new deletion rule)

R is again a symmetric predicate.



¬Pzz
$\{y \leftarrow x, z \leftarrow x\}$
Pxy
Rxy
$\sigma_1 = \{x \leftarrow a, y \leftarrow v\}$
$\sigma_2 = \{x \leftarrow v, y \leftarrow a\}$
¬Rav
$\sigma_1' = \{x' \leftarrow a, v \leftarrow a\}$
$\sigma_2' = \{x' \leftarrow a, v \leftarrow a\}$
Rx'x'

Let us assume that for some reason σ_2 was removed. If it is not physically destroyed, but only marked in some way and we are able to inherit σ_1 and σ_2 separately, it is possible to detect that the descedant σ_1' of σ_1 is an instance of the removed mgu σ_2'. Now we can remove σ_1' too and with σ_1' we can remove the whole link. (The completeness proof for this deletion rule is not part of this report and has to be given elsewhere.) ■

In this report a method is presented to inherit individual unifiers for an extensive class of theory unification algorithms and arbitrary literal-link types (links connecting literals, not terms).

Throughout the paper we use the following standard mathematical notation:

$\in$	set membership	$\setminus$	set difference
$\cup$	set union	$\cap$	set intersection
$\subseteq$	subset relation	$\lvert M\rvert$	cardinality of set M
$\emptyset$	empty set		
$\exists$	existential quantifier	$\forall$	universal quantifier
$\wedge$	logical and	$\vee$	logical or
$\neg$	negation	$\Rightarrow$	implication
■	end of an example, definition or proof	□	end of case in a proof by cases
$f\rvert_M$	function f restricted to a subset of its domain.	DOM(f)	Domain of the function f

2. Basic Notions

We use the standard notions of first order logic: Given pairwise disjoint alphabets, let $\mathbf{V}$ be the set of ***variable*** symbols; $\mathbf{F}_n$ be the set of n-ary ***function*** symbols, $\mathbf{F}_0 = \mathbf{C}$ the set of ***constant*** symbols and $\mathbf{F} = \bigcup \mathbf{F}_n$. Let $\mathbf{P}_n$ be the set of n-ary ***predicate*** symbols and $\mathbf{P} = \bigcup \mathbf{P}_n$. Furthermore let $\mathbf{T}$ be the set of terms, i.e $\mathbf{T}$ is the smallest set with $\mathbf{V} \subseteq \mathbf{T}$ and $f(t_1, \ldots, t_n) \in \mathbf{T}$ for all $f \in \mathbf{F}_n$ and $t_i \in \mathbf{T}$. If $P \in \mathbf{P}_n$ and $t_i \in \mathbf{T}$ we call $P(t_1, \ldots, t_n)$ an ***atom***. ***Literals*** are signed atoms (+A resp. -A) and $\mathbf{L}$ is the set of all literals. ***Clauses*** are sets of literals.

For any object t containing variables we define $\mathbf{V}(t)$ as the set of all ***variables occurring in*** t and $\mathbf{V}(t_1, \ldots, t_n)$ as an abbreviation for $\mathbf{V}(t_1) \cup \ldots \cup \mathbf{V}(t_n)$.

Substitutions

A ***substitution*** σ is an endomorphism on the term algebra associated with $\mathbf{T}$ which is identical almost everywhere.

Each substitution σ is completely determined by its restriction $\sigma|_{\mathbf{V}}$ [He83, I.2]. We will make frequent use of this property, writing $\{x_1 \leftarrow t_1, \ldots, x_n \leftarrow t_n\}$ for instance to represent a substitution σ with $\sigma(x_i) = t_i$, $1 \le i \le n$. For such a substitution σ we define:
$\mathrm{VARS}(\sigma) := (x_1, \ldots, x_n)$ and $\mathrm{TERMS}(\sigma) := (t_1, \ldots, t_n)$

ε is the identity substitution and $\mathbf{\Sigma}$ is the set of all substitutions.

The ***codomain*** of a substitution σ is $\mathrm{COD}(\sigma) := \sigma(\mathrm{DOM}(\sigma))$
The set of ***variables introduced by*** σ is defined by $\mathrm{VCOD}(\sigma) := \mathrm{V}(\mathrm{COD}(\sigma))$
and the set of ***variables of*** σ is $\mathbf{V}(\sigma) := \mathrm{DOM}(\sigma) \cup \mathrm{VCOD}(\sigma)$

For two substitutions $\sigma, \mu \in \mathbf{\Sigma}$ we write $\sigma\mu$ for the usual functional composition and $\sigma|_V$ for the restriction of σ to the set of variables V. Substitutions may also be applied to atoms (and consequently to literals and clauses too): $\sigma P(t_1, \ldots, t_n) := P(\sigma t_1, \ldots, \sigma t_n)$.

$\mathbf{\Sigma}^* := \{\sigma \in \mathbf{\Sigma} \mid \sigma\sigma = \sigma\}$ is the set of ***idempotent substitutions***.

We mention two useful properties of idempotent substitutions which can be found in [He83]:

2.1 Lemma

Let σ be a substitution and t be a term. Then

a) σ is idempotent iff $\mathrm{DOM}(\sigma) \cap \mathrm{VCOD}(\sigma) = \emptyset$. (Lemma I.5 in [He83])
b) If σ is idempotent then $\mathrm{DOM}(\sigma) \cap \mathbf{V}(\sigma t) = \emptyset$. (Lemma I.4 in [He83]) ■

Sorted Logic

In order to cover many variants of sorted logics, we define the notion of a stable sort calculus.

2.2 Definition (stable sort calculus)

We call a given many-sorted calculus a ***stable sort calculus*** if it defines the notions of sort symbols $\mathbf{S}$, a sort function $[...]$ and a predicate $\mathbf{WS}$ satisfying the following properties:

a) $\mathbf{S}$ is a finite non-empty set of sort symbols with a partial order $\le$, the subsort order, imposed on it.

b) $[...] : \mathbf{T} \to \mathbf{S}$ is a sort function satisfying the properties:

i) $\mathbf{V} \cup \mathbf{C} \subseteq \mathrm{DOM}([...])$

ii) $t = f(t_1, \dots, t_n) \in \mathrm{DOM}([...]) \Rightarrow t_i \in \mathrm{DOM}([...])\ 1 \le i \le n$ and

$\forall \sigma \in \mathbf{\Sigma}$, $(\forall x \in \mathbf{V}\ [\sigma x] \le [x]) \Rightarrow [\sigma t] \le [t]$

(The sort of an instantiated term is smaller or equal to the sort of the term).

c) $\mathbf{WS}$ is a meta-predicate for terms and literals with the properties:

i) If A is a term then $\mathbf{WS}(A) \Leftrightarrow A \in \mathrm{DOM}([...])$.

ii) If $A = \pm P(t_1, \dots, t_n)$ is a literal then $\mathbf{WS}(A)$ implies $\mathbf{WS}(t_i)$ $1 \le i \le n$ and

$\forall\ s_i \in \mathbf{T}\ [s_i] \le [t_i] \Rightarrow \mathbf{WS}(\pm P(s_1, \dots, s_n))$. ■

$\mathbf{WS}$ selects terms and literals with a property usually called "well sorted". Therefore let $\mathbf{WST} := (t \in \mathbf{T} |\ \mathbf{WS}(t))$ and $\mathbf{WSL} := (L \in \mathbf{L} |\ \mathbf{WS}(L))$ be the sets of well sorted terms resp. literals, $\mathbf{WS\Sigma} = (\sigma \in \mathbf{\Sigma} |\ \forall\ x \in \mathbf{V}\ [\sigma(x)] \le [x])$ be the set of well sorted substitutions and $\mathbf{WS\Sigma}^* = \mathbf{WS\Sigma} \cap \mathbf{\Sigma}^*$. (We write $\mathbf{WS}(\sigma)$ if $\sigma \in \mathbf{WS\Sigma}$.)

The notion of a stable sort calculus covers all the many-sorted calculi where instances of well sorted terms are again well sorted terms. (see for example the ΣRP calculus of Ch. Walther [Wa82a] or the ΣRP* calculus of M. Schmidt-Schauss [SS85]). A calculus where for example x^{-1} is regarded as well sorted, but 0^{-1} not, is not of this type.

Throughout the rest of this paper we consider only stable sort calculi.

2.3 Lemma

Let σ be a well sorted substitution. Then

a) If t is a well sorted term or literal, then σt is well sorted, too.

b) If t is a well sorted term then $[\sigma t] \le [t]$.

c) The composition of two well sorted substitutions is again a well sorted substitution.

The proofs follow immediately from the definition of a well sorted substitution and Def. 2.2,b,ii and c,ii. ■

Renaming Substitutions

Renaming substitutions are substitutions mapping variables into variables of the same sort. They are defined as follows:

2.4 Definition (Renaming substitutions)

An idempotent substitution $\varrho \in \mathbf{WS\Sigma}^*$ is called a ***renaming substitution*** for the variables $W \neq \emptyset$ iff

a) $DOM(\varrho) = W$

b) $\varrho W \subseteq \mathbf{V}$

c) $[x] = [\varrho x]$ for every variable x.

d) ϱ is injective on W, i.e $x \neq y \Rightarrow \varrho x \neq \varrho y$. ■

For a set of variables $W \subseteq \mathbf{V}$ we write:

$REN_{=}(W)$ for the set of all renaming substitutions ϱ with $DOM(\varrho) = W$ and

$REN_{\subseteq}(W)$ for the set of all renaming substitutions ϱ with $DOM(\varrho) \subseteq W$.

Furthermore we use $REN_{\sim}(O)$ as an abbreviation for $REN_{\sim}(\mathbf{V}(O))$ for any object O containing variables and $\sim \in \{=, \subseteq\}$.

2.5 Lemma

If ϱ is a renaming substitution then $VCOD(\varrho) = COD(\varrho)$.

The lemma follows immediately from $VCOD(\varrho) \subseteq \mathbf{V}$ and the injectivity condition for renaming substitutions. ■

The Converse of a Renaming Substitution

2.6 Definition (Converse of a renaming substitution)

Let $\varrho \in REN_{=}(V)$, $V \subseteq \mathbf{V}$ be a renaming substitution. We define the ***converse*** ϱ^c of ϱ by: $\quad \varrho^c x = y \quad$ iff $\quad \varrho y = x \quad$ for each $x \in V$. ■

An example: $\quad \varrho = \{x \leftarrow y, z \leftarrow u\} \quad \Rightarrow \quad \varrho^c = \{y \leftarrow x, u \leftarrow z\}$

That the converse of a renaming substitution is (unfortunately) not the inverse, but something similar is shown in lemma 2.8. First we need some trivial properties of the converse:

2.7 Lemma

Let ϱ be a renaming substitution. Then

a) $VCOD(\varrho^c) = DOM(\varrho)$ and

b) $DOM(\varrho^c) = VCOD(\varrho)$ and

c) ϱ^c is idempotent.

Proof

a)
$$\begin{aligned} VCOD(\varrho^c) &= \mathbf{V}(\varrho^c\, DOM(\varrho^c)) \\ &= \{\varrho^c x \mid \varrho^c x = y, y \neq x\} \\ &= \{\, y \mid \varrho y = x, y \neq x\} && \text{(Def. 2.6)} \\ &= DOM(\varrho) && \square \end{aligned}$$

b)
$$\begin{aligned} DOM(\varrho^c) &= \{x \mid \varrho^c x = y, y \neq x\} \\ &= \{x \mid \varrho y = x, y \neq x\} && \text{(Def. 2.6)} \\ &= COD(\varrho) && \square \end{aligned}$$

c) from a) and b) together with 2.4 we know that ϱ^c is a substitution $\varrho^c \in \Sigma$.

$\varrho \in \Sigma^* \Rightarrow \mathrm{VCOD}(\varrho) \cap V = \emptyset$

$\Rightarrow \mathrm{VCOD}(\varrho) \cap \mathrm{DOM}(\varrho) = \emptyset$ (because $\varrho \in \mathrm{REN}_=(V)$)

$\Rightarrow \mathrm{COD}(\varrho) \cap \mathrm{DOM}(\varrho) = \emptyset$ (Lemma 2.5)

$\Rightarrow \mathrm{DOM}(\varrho^c) \cap \mathrm{VCOD}(\varrho^c) = \emptyset$ (a, b)

$\Rightarrow \varrho^c$ is idempotent. (Lemma 2.1) ■

<u>Corollary</u> $\varrho \in \mathrm{REN}_=(V) \Leftrightarrow \varrho^c \in \mathrm{REN}_=(\mathrm{COD}(\varrho))$

<u>2.8 Lemma</u>

If ϱ is a renaming substitution, then $\varrho\varrho^c = \varrho$ and $\varrho^c\varrho = \varrho^c$

The proofs are given in [He83], Lemma 1.12.

<u>Permutations</u>

A substitution ζ is a permutation if there exists a ζ^- such that $\zeta\zeta^- = \varepsilon$.

($\zeta = \{x \leftarrow y, y \leftarrow z, z \leftarrow x\}$ for example is a permuation with $\zeta^- = \{x \leftarrow z, z \leftarrow y, y \leftarrow x\}$).

In the inheritance mechanisms, defined in chapter 6, we need permutations consisting of one cycle of length 2. They look like $\hat{\varrho} = \{x \leftarrow y, y \leftarrow x, u \leftarrow v, v \leftarrow u\}$ and are constructed from ordinary renaming substitutions ϱ as follows:

$$\hat{\varrho}\,x = \begin{cases} \varrho x & \text{if } x \in \mathrm{DOM}(\varrho) \\ \varrho^c x & \text{otherwise.} \end{cases}$$

<u>2.9 Lemma</u> (some properties of $\hat{\varrho}$)

Let ϱ be a renaming substitution.

a) $\hat{\varrho}|_{\mathrm{DOM}(\varrho)} = \varrho$

b) $\hat{\varrho}\hat{\varrho} = \varepsilon$ ($\Rightarrow \hat{\varrho}$ is a permutation)

c) $\hat{\varrho}\varrho|_{\mathrm{DOM}(\varrho)} = \varepsilon$

<u>Proof</u>

a) Follows immediately from the definition. □

b) $\mathrm{DOM}(\hat{\varrho}) = \mathrm{DOM}(\varrho) \cup \mathrm{DOM}(\varrho^c)$ (by the definition)

$= \mathrm{DOM}(\varrho) \cup \mathrm{COD}(\varrho)$ (Lemma 2.7,b)

$= \mathrm{DOM}(\varrho) \cup \mathrm{VCOD}(\varrho)$ (Lemma 2.5)

As ϱ is idempotent we know that $\mathrm{DOM}(\varrho) \cap \mathrm{VCOD}(\varrho) = \emptyset$. Therefore we can split $\mathrm{DOM}(\hat{\varrho})$ into the two disjoint sets $\mathrm{DOM}(\varrho)$ and $\mathrm{VCOD}(\varrho)$. Now let $y \in \mathrm{DOM}(\hat{\varrho})$.

<u>Case 1:</u> $y \in \mathrm{DOM}(\varrho)$ with $\varrho y = x$

$\hat{\varrho}\hat{\varrho}y = \hat{\varrho}\varrho y$ (Def. and $y \in \mathrm{DOM}(\varrho)$)

$= \hat{\varrho}x$

$= \varrho^c x$ ($x \in \mathrm{COD}(\varrho) = \mathrm{DOM}(\varrho^c)$)

$= y$ (Def. 2.6) □

<u>Case 2:</u> $y \in \mathrm{VCOD}(\varrho) = \mathrm{DOM}(\varrho^c)$ with $\varrho^c(y) = x$

$\hat{\varrho}\hat{\varrho}y = \hat{\varrho}\,\varrho^c y$ (Def. and $y \in \mathrm{DOM}(\varrho^c)$)

$= \hat{\varrho}\,x$

$= \varrho x$ ($x \in \mathrm{COD}(\varrho^c) = \mathrm{DOM}(\varrho)$)

$= y$ (Def. 2.6) □

Case 1 + Case 2 $\Rightarrow \hat{\varrho}\hat{\varrho} = \varepsilon$ □

c) see proof b), case 1. ■

Renaming Operations on Substitutions

For the inheritance mechanisms of chapter 6 we define a renaming operation on substitutions. If for example $\sigma = \{x \leftarrow fy, z \leftarrow gav\}$ is a substitution and $\varrho = \{y \leftarrow w, z \leftarrow u\}$ is a renaming substitution, then the ***application of*** ϱ to σ: $\varrho \bullet \sigma = \{x \leftarrow fw, u \leftarrow gav\}$ is a new substitution which is different from the functional composition
$\varrho\,\sigma = \{x \leftarrow fw, z \leftarrow gav, y \leftarrow w\}$

2.10 Definition (Renaming of a substitution)

For a substitution $\sigma \in \mathbf{\Sigma}$ and a renaming substitution ϱ we define a renaming operation $\bullet$: $\quad \varrho \bullet \sigma := \hat{\varrho}\,\sigma\,\hat{\varrho}$ ■

Some useful properties of the renaming operation are listed in the next four lemmas.

2.11 Lemma

If ϱ is a renaming substitution then $\mathrm{DOM}(\varrho \bullet \sigma) = \hat{\varrho}\mathrm{DOM}(\sigma)$.

Proof

$\varrho \bullet \sigma x = \hat{\varrho}\,\sigma\,\hat{\varrho}x \neq x \Leftrightarrow \sigma\,\hat{\varrho}x \neq \hat{\varrho}x \Leftrightarrow \hat{\varrho}x \in \mathrm{DOM}(\sigma) \Leftrightarrow \hat{\varrho}\hat{\varrho}x = x \in \hat{\varrho}\mathrm{DOM}(\sigma)$. ■

The following lemma shows that the definition of $\bullet$ in fact reflects our intuition of a renaming of a substitution:

2.12 Lemma

If $\sigma = \{x_1 \leftarrow t_1, \ldots, x_n \leftarrow t_n\} \in \mathbf{\Sigma}$ and $\varrho \in \mathrm{REN}_{\sqsubseteq}(\sigma)$ with $\mathrm{COD}(\varrho) \cap \mathbf{V}(\sigma) = \emptyset$, then

$$\begin{aligned}\varrho \bullet \sigma &= \{\varrho x_1 \leftarrow \varrho t_1, \ldots, \varrho x_n \leftarrow \varrho t_n\}\\ &= \{\varrho x_1 \leftarrow \varrho\sigma x_1, \ldots, \varrho x_n \leftarrow \varrho\sigma x_n\}\end{aligned}$$

Proof

With lemma 2.11 and $\mathrm{COD}(\varrho) \cap \mathbf{V}(\sigma) = \emptyset$ we have $\mathrm{DOM}(\varrho \bullet \sigma) = \varrho\mathrm{DOM}(\sigma)$. Therefore let x be a variable and $x \in \varrho\mathrm{DOM}(\sigma)$ (nontrivial case), then by the injectivity of ϱ and $\mathrm{COD}(\varrho) \cap \mathbf{V}(\sigma) = \emptyset$ there exists a k, $1 \leq k \leq n$ with $x_k \in \mathrm{DOM}(\sigma)$ and $\varrho x_k = x$. With 2.6 we have now: $x_k = \varrho^c x$ and therefore it is $\varrho \bullet \sigma x = \hat{\varrho}\sigma\hat{\varrho}x = \varrho\sigma\varrho^c x = \varrho\sigma x_k = \varrho t_k$. ■

2.13 Lemma

The renaming operation preserves the idempotency and well-sortedness of a substitution.

Proof

$\sigma\sigma = \sigma \Rightarrow (\varrho \bullet \sigma)(\varrho \bullet \sigma) = (\hat{\varrho}\sigma\hat{\varrho})(\hat{\varrho}\sigma\hat{\varrho}) = \hat{\varrho}\,\sigma\sigma\,\hat{\varrho} = \hat{\varrho}\sigma\hat{\varrho} = \varrho \bullet \sigma \quad \Rightarrow \quad \varrho \bullet \sigma \in \mathbf{\Sigma}^*$

The well-sortedness is preserved because ϱ renames variables by other variables of the <u>same</u> sort (Def. 2.4,c). ■

The next lemma shows that the renaming of an instantiated term is equivalent to the instantiation of the renamed term with the renamed substitution. This result will frequently be used in the proofs of chapter 6.

2.14 Lemma

Let t be a term, $\sigma \in \mathbf{\Sigma}$ and ϱ be a renaming substitution, then we have

a) $(\varrho \bullet \sigma)\hat{\varrho} = \hat{\varrho}\sigma$ and

b) $\varrho \in \mathrm{REN}_{\sqsubseteq}(\mathbf{V}(t,\sigma))$ and $\mathrm{VCOD}(\varrho) \cap \mathbf{V}(t,\sigma) = \emptyset \Rightarrow (\varrho \bullet \sigma)\varrho t = \varrho\sigma t$

Proof:

a) $(\varrho \bullet \sigma)\hat{\varrho} = \hat{\varrho}\sigma\hat{\varrho}\hat{\varrho} = \hat{\varrho}\sigma$ (Lemma 2.9,b)

b) Follows form the definition of $\hat{\varrho}$ and a). ■

Corollary The statement of lemma 2.14 remains valid if t is a literal. ■

3. Congruence Relations and Unification

A unification problem in its most abstract form can be stated as follows [Sz82]:
Let S be a formal language containing variables, $\mathbf{\Sigma}$ the set of substitutions in S, and $\approx \subseteq S\times S$ a relation. For any $p,q \in S$ a substitution $\sigma \in \mathbf{\Sigma}$ with $\sigma(p) \in S$, $\sigma(q) \in S$ and $\sigma(p) \approx \sigma(q)$ is called a $\approx$-unifying substitution ($\approx$-unifier).
Unification theory investigates the computability and representability of such $\approx$-unifiers.

This paper is concerned with two such languages and relations, first the language of well **sorted terms WST** with a relation $\approx$ satisfying special properties and then the language of **literals** with an extension of $\approx$. The relation between terms $\approx$ is usually [Si84] defined by an equational theory and most of the known unification algorithms work for special equational theories. For the solution of the inheritance problem to be addressed in this paper, it is not necessary however for $\approx$ to be an equationally defined relation. $\approx$ may remain an abstract relation and the unification algorithm need not be defined explicitly but has to fulfill certain requirements.

3.1 Definition (Congruence relations)

A ***stable congruence relation*** $\approx$ on well sorted terms is a relation satisfying the following properties:

a) $\approx$ is an equivalence relation.

b) congruence: Let t_i, s_i with $t_i \approx s_i$, $1 \le i \le n$ be pairs of terms.
If $f(t_1, \dots, t_n)$ and $f(s_1, \dots, s_n)$ are well sorted, then $f(t_1, \dots, t_n) \approx f(s_1, \dots, s_n)$.

c) stability: If $s,t \in \mathbf{WST}$ are two terms and $\sigma \in \mathbf{WS\Sigma}$ and $s \approx t$ then $\sigma s \approx \sigma t$.

3.2 An Example

Assume $\langle \mathbf{S},[\dots],\mathbf{WS}\rangle$ is the ΣRP calculus of Ch. Walther [Wa82a] with $\mathbf{S}=(A,B)$ and $A<B$. If we have a constant a of sort A and two functions $g{:}A\to A$ and $f{:}B\times B\to B$ and $\approx$ is the relation expressing the idempotence of f: $(fxx=x)$. Then we have $a \approx faa$, but not $ga \approx g(faa)$, because $g(faa)$ is not a well sorted term. If, however, we consider the ΣRP^* calculus of M. Schmidt-Schauss [SS85] and declare f being polymorphic:
$f \in \mathbf{F}_{((B,B,B)(A,B,B)(B,A,B)(A,A,A))}$ then $[faa]=A$ and $g(faa)$ is a well sorted term.
Now $g(a) \approx g(faa)$ holds. ■

Throughout this chapter let $\approx$ denote such a stable congruence.

The symbol $\approx$ is also used for the canonical extension of the $\approx$-relation to lists of terms:
$(t_1, \dots, t_n) \approx (s_1, \dots, s_n)$ iff $t_i \approx s_i$ for $1 \le i \le n$.

Extension of the $\approx$-Relation to Substitutions

There are two possible ways to extend the $\approx$-relation to substitutions:

3.3 Definition (strong extension of the $\approx$-relation to substitutions)

For two substitutions σ, $\mu \in \mathbf{WS\Sigma}$ and a set of variables W we write
$\sigma \approx \mu\,[W]$ iff for each variable $x \in W$: $\sigma x \approx \mu x$. ■

3.4 Lemma

$\approx [W]$ is an equivalence relation on well sorted substitutions.
The proof follows immediately from the equivalence condition 3.1 a). ■

3.5 Lemma (Correspondence between the $\approx$ relations on $\mathbf{WST}$ and $\mathbf{WS\Sigma}$.)

If t is a well sorted term and σ, μ are well sorted substitutions with $\sigma \approx \mu\ [\mathbf{V}(t)]$, then $\sigma t \approx \mu t$.

Proof by induction

The induction basis follows trivially from the definition of the $\approx$-relation.

Induction step

Let $t = f(t_1, \ldots, t_n) \in \mathbf{WST}$

$\Rightarrow\ \sigma t \in \mathbf{WST}$ and $\mu t \in \mathbf{WST}$ (Lemma 2.3)

$\Rightarrow\ \sigma t_i \approx \mu t_i$ (Induction hypotheses)

$\Rightarrow\ f(\sigma t_1, \ldots, \sigma t_n) \approx f(\mu t_1, \ldots, \mu t_n)$ (Congruence property 3.1,b)

$\Rightarrow\ \sigma f(t_1, \ldots, t_n) \approx \mu f(t_1, \ldots, t_n)$. ■

The following definition gives the second extension of $\approx$ to substitutions.

3.6 Definition (Subsumption relation on substitutions)

a) Let $\sigma, \mu \in \mathbf{WS\Sigma}$, $W \subseteq \mathbf{V}$ then $\sigma \le \mu\ [W]$ iff $\exists\ \lambda \in \mathbf{WS\Sigma}\quad \sigma \approx \lambda\mu\ [W]$

b) Let $\sigma, \mu \in \mathbf{WS\Sigma}$, $W \subseteq \mathbf{V}$ then $\sigma \equiv \mu\ [W]$ iff $\sigma \le \mu\ [W]$ and $\mu \le \sigma\ [W]$

3.7 Lemma

a) $\le [W]$ is a partial ordering on well sorted substitutions.

b) $\equiv [W]$ is an equivalence relation on well sorted substitutions.

Proof

a) Since the reflexivity follows trivially, the transitivity of $\le [W]$ remains to be shown: Let σ_1, σ_2 and $\sigma_3 \in \mathbf{WS\Sigma}$ with $\sigma_1 \le \sigma_2\ [W]$ and $\sigma_2 \le \sigma_3\ [W]$ for some variables W.

$\Rightarrow\ \exists\ \lambda_1, \lambda_2 \in \mathbf{WS\Sigma}$ with $\forall x \in W\ \sigma_1 x \approx \lambda_1 \sigma_2 x$ and $\sigma_2 x \approx \lambda_2 \sigma_3 x$

$\Rightarrow\ \forall x \in W\quad \sigma_1 x \approx \lambda_1 \lambda_2 \sigma_3 x$ (Def. 3.1,a,c and Lemma 2.3,c)

$\Rightarrow\ \sigma_1 \le \sigma_3\ [W]$ □

b) The reflexivity and symmetry are trivial and the transitivity follows from the transitivity of $\le [W]$. ■

3.8 Lemma (Correspondence between the $\le$ and $\equiv$-relations and the $\approx$-relation on $\mathbf{WST}$)

Let s,t be well sorted terms and σ a substitution with $\sigma s \approx \sigma t$.

If $\tau \in \mathbf{WS\Sigma}$ with $\tau \le \sigma\ [\mathbf{V}(s,t)]$ then $\tau s \approx \tau t$.

Proof

$\tau \le \sigma\ [\mathbf{V}(s,t)] \Rightarrow\ \exists\ \lambda \in \mathbf{WS\Sigma}\quad \tau \approx \lambda\sigma\ [\mathbf{V}(s,t)]$

$\Rightarrow\ \tau s \approx \lambda\sigma s$ (Lemma 3.5)

$\approx \lambda\sigma t$ (stability of $\approx$)

$\approx \tau t$ (Lemma 3.5)

■

Since the same lemma holds for the $\equiv$-relation as well, the $\equiv$-relation just classifies the substitutions which can be used alternatively as unifiers for two given terms.

The next lemma is very instructive. It shows that the definition of the ≡-relation on substitutions allows to construct equivalent idempotent substitutions for each $\sigma \in \mathbf{WS\Sigma}$.

3.9 Lemma

For every well sorted substitution σ there exists an idempotent substitution σ' with $\sigma \equiv \sigma'$ [DOM(σ)].

Proof

Let $\varrho \in \mathrm{REN}_=(\mathrm{DOM}(\sigma) \cap \mathrm{VCOD}(\sigma))$ such that $\mathrm{VCOD}(\varrho) \cap \mathbf{V}(\sigma) = \emptyset$

$\Rightarrow \mathrm{DOM}(\varrho^c) \cap \mathbf{V}(\sigma) = \emptyset$. (Lemma 2.5, 2.7,b)

Let $\sigma' = \varrho\sigma|_{\mathrm{DOM}(\sigma)}$.

$\Rightarrow \mathrm{DOM}(\sigma') = \mathrm{DOM}(\sigma)$ and $\mathrm{VCOD}(\sigma') \subseteq (\mathrm{VCOD}(\sigma) \setminus \mathrm{DOM}(\sigma)) \cup \mathrm{VCOD}(\varrho)$.

Both sets are disjoint and hence by lemma 2.1 σ' is idempotent.

Furthermore we have $\sigma' \leq \sigma$ [DOM(σ)] because $\sigma' \approx \varrho\,\sigma$ [DOM(σ)] and

$\sigma \leq \sigma'$ [DOM(σ)] because $\sigma \approx \varrho^c\sigma$ [DOM(σ)]

$\approx \varrho^c\varrho\sigma$ [DOM(σ)] (Lemma 2.8)

$\approx \varrho^c\sigma'$ [DOM(σ)]

$\Rightarrow \sigma \equiv \sigma'$ [DOM(σ)]. ■

The proof shows how to make a substitution idempotent by a renaming of the critical variables in the codomain. Whenever we postulate the existence of a substitution which needs to be determined only up to the ≡-relation, we can therefore assume that this substitution is idempotent (see for instance the definition of unifiers below).

Every symmetric and transitive relation $\sim\ \subseteq M \times M$ for a certain set M can easily be extended to sets of elements of M if we demand the existence of a bijection between these sets. We define this extension for arbitrary sets M and ~-relations, but we use it especially for sets of substitutions and the ≡ [W] relation.

3.10 Definition

Let $\sim\ \subseteq M \times M$ be a symmetric and transitive relation for a set M and $\Sigma_1, \Sigma_2 \subseteq M$. We say $\Sigma_1 \sim \Sigma_2$ iff there exists a bijection ψ: $\Sigma_1 \rightarrow \Sigma_2$ such that $\forall \sigma \in \Sigma_1\ \ \sigma \sim \psi(\sigma)$. ■

3.11 Lemma (criterion for the equivalence of such sets.)

Let $\sim\ \subseteq M \times M$ be a symmetric and transitive relation for a set M and $\Sigma_1, \Sigma_2 \subseteq M$ be minimal with respect to ~, i.e. $\sigma_1 \neq \sigma_2 \in \Sigma_i \Rightarrow \sigma_1 \nsim \sigma_2$ for $i = 1,2$,

then: $\Sigma_1 \sim \Sigma_2$ iff

a) $\forall \sigma \in \Sigma_1\ \exists \tau \in \Sigma_2\ \sigma \sim \tau$ and b) $\forall \tau \in \Sigma_2\ \exists \sigma \in \Sigma_1\ \tau \sim \sigma$

Proof

"$\Rightarrow$" for $\sigma \in \Sigma_1$ select $\tau = \psi(\sigma) \in \Sigma_2$ and for $\tau \in \Sigma_2$ select $\sigma = \psi^{-1}(\tau) \in \Sigma_1$.

"$\Leftarrow$" Let $\psi \subseteq \Sigma_1 \times \Sigma_2$ and $(\sigma,\tau) \in \psi$ iff $\sigma \sim \tau$.

We have to show that ψ is a bijection.

ψ is surjective by the definition. Now let $(\sigma,\tau_1) \in \psi$ and $(\sigma,\tau_2) \in \psi$.

$\Rightarrow \sigma \sim \tau_1$ and $\sigma \sim \tau_2$

$\Rightarrow \tau_1 \sim \sigma$ (symmetry of ~)

$\Rightarrow \tau_1 \sim \tau_2$ (transitivity of ~)

$\Rightarrow \tau_1 = \tau_2$ (minimality of Σ_2)

$\Rightarrow \psi$ is a function.

The injectivity is shown analogously.

$\Rightarrow \psi$ is a bijection

$\Rightarrow \Sigma_1 \sim \Sigma_2$ ■

3.12 Lemma (Independence of the $\equiv [W]$ relation under renaming)

Let $W \subseteq \mathbf{V}$, $\Sigma_1, \Sigma_2 \subseteq \mathbf{WS\Sigma}$. Then $\Sigma_1 \equiv \Sigma_2 [W] \Rightarrow \rho \bullet \Sigma_1 \equiv \rho \bullet \Sigma_2 [\hat{\rho}(W)]$

Proof

The main idea of the proof is to show that for two well sorted substitutions σ, τ:

$\sigma \equiv \tau [W] \Rightarrow \rho \bullet \sigma \equiv \rho \bullet \tau [\hat{\rho}(W)]$:

$\sigma \equiv \tau [W] \Rightarrow \exists \lambda_1, \lambda_2 \forall x \in W \;\; \sigma(x) \approx \lambda_1 \tau(x)$ and $\tau(x) \approx \lambda_2 \sigma(x)$.

Now let $y \in \hat{\rho}(W)$.

$\Rightarrow \exists x \in W \;\; y = \hat{\rho}(x)$ and $x = \hat{\rho}(y)$

$\Rightarrow \sigma\hat{\rho}(y) = \sigma(x) \approx \lambda_1 \tau(x) = \lambda_1 \tau \hat{\rho}(y)$

$\Rightarrow \hat{\rho}\sigma\hat{\rho}(y) \approx \hat{\rho}\, \lambda_1 \tau \hat{\rho}(y)$ (stability of $\approx$)

$\Rightarrow \hat{\rho}\sigma\hat{\rho}(y) \approx \hat{\rho}\, \lambda_1 \hat{\rho}\hat{\rho} \tau \hat{\rho}(y)$ ($\hat{\rho}\hat{\rho} = \varepsilon$)

$\Rightarrow \rho \bullet \sigma \approx (\rho \bullet \lambda_1)(\rho \bullet \tau) [\hat{\rho}(W)]$

$\Rightarrow \rho \bullet \sigma \leq \rho \bullet \tau [\hat{\rho}(W)]$.

The other case works in a similar way.

$\Rightarrow \rho \bullet \sigma \equiv \rho \bullet \tau [\hat{\rho}(W)]$

The theorem now follows immediately. ■

The definition of a congruence relation for terms is now extended in two steps to a relation on literals.

3.13 Definition (extension of the $\approx$ relation to literals)

a) (canonical extension of $\approx$)

Let L and K be terms or literals

We define: $L \simeq K$ iff the two signs and predicates are equal and the $\approx$-relation holds for the two termlists

b) ("natural" extension of $\approx$)

A given symmetric and stable relation $\sim$ between literals can be combined with the $\simeq$-relation to a new relation $\approx \; = \; \simeq \circ \sim$ on literals. We call it a (compatible) ***extension*** of the $\approx$ relation (on well sorted terms). Its properties are:

i) $\forall L,K \in \mathbf{WFL} \;\; L \approx K \Leftrightarrow \exists R \in \mathbf{WFL} \;\; L \simeq R$ and $R \sim K$ ($\approx \; = \; \simeq \circ \sim$ "compatibility")

ii) $\forall L,K \in \mathbf{WFL}, \sigma, \in \mathbf{WF\Sigma} \;\; L \approx K \Rightarrow \sigma(L) \approx \sigma(K)$ (stability)

Remarks

-- Definition a) is the canonical extension of $\approx$ to literals which observes the properties of terms represented by $\approx$. The $\sim$-relation described in b) may take properties of predicates (symmetry etc.) and the sign of the literal into account. This relation needs neither be reflexive nor transitive. Finally the $\approx$-relation on literals combines the properties of predicates and terms. It may for instance describe "complementarity of literals" (For example $+P(a, f(b,c)) \approx -P(f(c,b), a)$ if P is symmetric and f commutative).

-- We also use the symbol $\approx$, which has originally been used for the relation between terms, for the relation between literals, because this relation is in some sense a "natural" extension. It allows to handle properties of literals in the same way as properties of terms. (The symmetry of a predicate symbol, for instance, and the commutativity of a symbol are not really very different properties). Furthermore it specifies the unification problem $\langle s \approx t \rangle$ for two terms resp. $\langle L \approx K \rangle$ for two literals and many lemmas can be proved once for both relations.

3.14 Lemma (another formulation for the compatibility property)

If $L,K \in \mathbf{WSL}$, $\sigma,\tau \in \mathbf{WS\Sigma}$ with $\sigma(L) \approx \sigma(K)$ and $\tau \approx \sigma [\mathbf{V}(L,K)]$ then $\tau(L) \approx \sigma(K)$ and $\tau(L) \approx \tau(K)$

Proof

$\tau \approx \sigma [\mathbf{V}(L,K)] \Rightarrow \sigma(K) \simeq \tau(K)$ and $\sigma(L) \simeq \tau(L)$

(trivial extension of lemma 3.5 to literals)

$\sigma(L) \approx \sigma(K) \Rightarrow \exists R \; \sigma(L) \simeq R \wedge R \sim \sigma(K)$

$\Rightarrow R \propto \tau(L)$ (transitivity of $\propto$)

$\Rightarrow \exists R\ \tau(L) \propto R \wedge R \sim \sigma(K)$

$\Rightarrow \tau(L) \approx \sigma(K)$ (symmetry)

$\Rightarrow \exists R\ \sigma(K) \propto R \wedge R \sim \tau(L)$

$\Rightarrow \tau(K) \propto R$

$\Rightarrow \tau(K) \approx \tau(L) \quad \Rightarrow \tau(L) \approx \tau(K)$ ■

3.15 Lemma (extension of lemma 3.8 to literals)

Let s and t be well sorted terms or literals and $\sigma \in \mathbf{WS\Sigma}$ with $\sigma(s) \approx \sigma(t)$.

If $\tau \in \mathbf{\Sigma}$ with $\tau \leq \sigma\,[\mathbf{V}(s,t)]$, then $\tau(s) \approx \tau(t)$.

Proof

$\tau \leq \sigma\,[\mathbf{V}(s,t)] \Rightarrow \exists \lambda \in \mathbf{WS\Sigma}\ \ \tau \approx \lambda\sigma\,[\mathbf{V}(s,t)]$

The first case, $s,t \in \mathbf{WST}$ is proved by lemma 3.8, therefore let $s,t \in \mathbf{WSL}$.

$\sigma(s) \approx \sigma(t) \Rightarrow \lambda\sigma(s) \approx \lambda\sigma(t)$ (stability)

$\Rightarrow \tau(s) \approx \tau(t)$ (Lemma 3.14)

■

Unifiers

Now we can come back to the main subject of this chapter: unification.

3.16 Definition (complete set of unifiers)

For arbitrary terms $s,t \in \mathbf{WST}$ or literals $s,t \in \mathbf{WSL}$, $U\Sigma(s,t) \subseteq \mathbf{WS\Sigma}^*$ is called a correct and complete set of unifiers ($\approx$-unifiers) for s and t iff

a) $\forall \sigma \in U\Sigma(s,t)\ \ \sigma(s) \approx \sigma(t)$ (correctness)

b) $\forall \sigma \in \mathbf{\Sigma}\ \ \sigma(s) \approx \sigma(t) \Rightarrow \exists\ \tau \in U\Sigma(s,t)\ \ \sigma \leq \tau\,[\mathbf{V}(s,t)]$ (completeness)

In addition we need a purly technical condition which makes the definition insensible against variable renamings and prevents such undesired things as $U\Sigma(x,y) = \{x \leftarrow y\}$ but $U\Sigma(x',y') = \{y' \leftarrow x'\}$

c) $\forall \rho \in REN_{\mathbf{C}}(\mathbf{V})\quad U\Sigma(\rho(s),\rho(t)) = \rho \bullet U\Sigma(s,t)$

An actual implementation of a unification algorithm usually fulfills this condition. To violate them would mean to put extra code into the algorithm which for instance sorts the variables according to some obscure ordering.

A complete set of unifiers $U\Sigma(s,t)$ is ***minimal*** if in addition

d) $\forall \sigma_1, \sigma_2 \in U\Sigma(s,t)\ \ \sigma_1 \leq \sigma_2\,[\mathbf{V}(s,t)] \Rightarrow \sigma_1 = \sigma_2$ and

e) $\forall \sigma \in U\Sigma(s,t)\ \ DOM(\sigma) \subseteq \mathbf{V}(s,t)$ hold. ■

This set is unique up to the $\equiv [\mathbf{V}(s,t)]$ relation and is called a most general set of unifiers. (It is usually written $\mu U\Sigma(s,t)$).

In order to calculate most general sets of unifiers form complete sets of unifiers, we define an instance removing function MAX:

3.17 Definition (Maximizer)

Let W be a set of variables. A function $MAX_W: 2^{\Sigma} \rightarrow 2^{\Sigma}$ with the following properties is called a Maximizer with respect to W:

a) $\forall \Sigma \subseteq \mathbf{WS\Sigma}, \sigma \in \Sigma\quad MAX_W(\Sigma) \subseteq \Sigma$ and $\exists \lambda \in MAX_W(\Sigma)\ \ \sigma \leq \lambda\,[W]$

b) $\forall \Sigma \subseteq \mathbf{WS\Sigma}, \forall \sigma_1, \sigma_2 \in MAX_W(\Sigma)\quad \sigma_1 \leq \sigma_2 [W] \Rightarrow \sigma_1 = \sigma_2$

c) $\forall \Sigma \subseteq \mathbf{WS\Sigma}, \forall \rho \in ren_{\mathbf{C}}(\mathbf{V})\quad MAX_W(\rho \bullet \Sigma) = \rho \bullet MAX_W(\Sigma)$

d) $\forall \sigma \in MAX_W(\Sigma)\ \ DOM(\sigma) \subseteq W$.

The function MAX_W (which needs not be unique) just removes all substitutions in Σ which are instances of other elements of Σ. It is insensible against variable renaming in the same sense as $U\Sigma$.

3.18 Lemma

For two well sorted terms or literals s and t: $\mu U\Sigma(s,t) := MAX_{\mathbf{V}(s,t)}(U\Sigma(s,t))$ is a set of most general unifiers for s and t.
The proof is obvious.

At this point it is necessary to put a further restriction onto the $\approx$ relation:
We consider only relations $\approx$ for which $\mu U\Sigma(s,t)$ exists and can be enumerated for every s and t, but $\mu U\Sigma(s,t)$ may be infinite.

The definitions for $U\Sigma$ and $\mu U\Sigma$ can be easily extended to termlists:
Let $t_1, \dots, t_n$ and $s_1, \dots, s_n \in \mathbf{WST}$ and $W = \bigcup \mathbf{V}(t_i,s_i)$
Then $U\Sigma((t_1, \dots, t_n), (s_1, \dots, s_n)) := \bigcap U\Sigma(t_i,s_i)$ and
$\mu U\Sigma((t_1, \dots, t_n), (s_1, \dots, s_n)) := MAX_W U\Sigma((t_1, \dots, t_n), (s_1, \dots, s_n))$

3.19 Lemma (Insensibility of $\mu U\Sigma$ against variable renaming)

Let s,t be well sorted terms or literals. Then
$\forall \rho \in REN_C(\mathbf{V}) \quad \mu U\Sigma(\rho(s),\rho(t)) = \rho \bullet \mu U\Sigma(s,t)$

The proof follows immediately from the appropriate conditions for $U\Sigma$ and MAX. ■

3.20 Lemma ("more general than most general is impossible.")

Let s,t be well sorted terms or literals and $\pi \in \mu U\Sigma(s,t)$;
let $\varphi \in \mathbf{\Sigma}$ with $\varphi(s) \approx \varphi(t)$ and $\pi \le \varphi$ $[\mathbf{V}(s,t)]$.
Then we have $\pi \equiv \varphi$ $[\mathbf{V}(s,t)]$

Proof

$\varphi(s) \approx \varphi(t) \Rightarrow \varphi|_{\mathbf{V}(s,t)}(s) \approx \varphi|_{\mathbf{V}(s,t)}(t)$

$\Rightarrow \exists \varphi' \in \mathbf{\Sigma}^* \; \varphi' \equiv \varphi$ $[\mathbf{V}(s,t)]$ (Lemma 3.9)

$\Rightarrow \exists \lambda \in \mu U\Sigma(s,t) \; \varphi' \le \lambda$ $[\mathbf{V}(s,t)]$ (Def. 3.17,a)

$\Rightarrow \pi \le \lambda$ $[\mathbf{V}(s,t)]$ (transitivity of $\le$)

$\Rightarrow \pi = \lambda$ (Def. 3.17,b)

$\Rightarrow \varphi' \le \pi$ $[\mathbf{V}(s,t)]$

$\Rightarrow \varphi \le \pi$ $[\mathbf{V}(s,t)]$

$\Rightarrow \varphi \equiv \pi$ $[\mathbf{V}(s,t)]$ (Def. 3.6,b)

■

3.21 Lemma

Let s and t be well sorted terms or literals and $\tau \in \mu U\Sigma(s,t) \neq \emptyset$;
let $\sigma \in \mathbf{\Sigma}$ with $\sigma(VARS(\tau)) \approx \sigma(TERMS(\tau))$ then we have $\sigma(s) \approx \sigma(t)$

Proof

$\sigma(VARS(\tau)) \approx \sigma(TERMS(\tau)) \Rightarrow \forall x \in \mathbf{V} \; \sigma(x) \approx \sigma\tau(x)$

$\Rightarrow \sigma \le \tau$ $[\mathbf{V}]$ $\Rightarrow \sigma(s) \approx \sigma(t)$ (Lemma 3.15) ■

4. The Basic Inheritance Theorem

Using the technical preparations of the last three chapters we are now able to prove a correspondence between unifiers of terms (or literals) and unifiers of their instances. This correspondence is the key for all inheritance algorithms in the following chapters.

For ease of notation we abbreviate:

$$\mu U\Sigma(\sigma, \Sigma)|_W := MAX_W(\bigcup_{\tau \in \Sigma} \{\pi|_W \mid \pi \in U\Sigma(\sigma(VARS(\tau)), \sigma(TERMS(\tau)))\})$$

4.1 The Inheritance Theorem

Let s and t be well sorted terms or literals and $\Theta = \mu U\Sigma(s,t) \neq \emptyset$;
let $\sigma \in \mathbf{WS\Sigma}$ with $\mathbf{V}(\sigma) \cap \mathbf{V}(\Theta) \subseteq \mathbf{V}(s,t)$ and let $W := \mathbf{V}(\sigma s, \sigma t)$;
then $\mu U\Sigma(\sigma s, \sigma t) \equiv \mu U\Sigma(\sigma, \theta)|_W\ [W]$

Proof

According to lemma 3.11 it is sufficient to prove

i) $\forall \pi \in \mu U\Sigma(\sigma s, \sigma t)\ \ \exists \varphi \in \mu U\Sigma(\sigma, \Theta)|_W$ with $\pi \equiv \varphi\ [W]$ and
ii) $\forall \varphi \in \mu U\Sigma(\sigma, \Theta)|_W\ \ \exists \pi \in \mu U\Sigma(\sigma s, \sigma t)$ with $\varphi \equiv \pi\ [W]$

Proof i)

Let $\pi \in \mu U\Sigma(\sigma s, \sigma t)$. Our goal is to find a $\varphi \in \mu U\Sigma(\sigma, \Theta)|_W$ with $\pi \equiv \varphi\ [W]$.

The idea is to extend π to a substitution φ^* which unifies $\sigma(VARS(\tau))$ and $\sigma(TERMS(\tau))$ for a $\tau \in \Theta$. Then we take a $\varphi' \in \mu U\Sigma(\sigma(VARS(\tau)),\sigma(TERMS(\tau)))$ which is more general than φ^* and show that $\varphi = \varphi'|_W$ unifies σs and σt. But since π is most general we can show that π is equivalent to φ.

With $\pi \in \mu U\Sigma(\sigma s, \sigma t)$
$\Rightarrow\ \pi\sigma s \approx \pi\sigma t$
$\Rightarrow\ \exists\, \tau \in \mu U\Sigma(s, t) \qquad \pi\sigma \leq \tau\ \ [\mathbf{V}(s,t)]$
$\Rightarrow\ \exists \lambda \in \mathbf{WS\Sigma} \qquad \pi\sigma \approx \lambda\tau\ [\mathbf{V}(s,t)]$ [*]

Now let $$\varphi^*(x) = \begin{cases} \lambda\tau(x) & x \in VCOD(\tau) \setminus \mathbf{V}(s,t) \\ \pi(x) & \text{otherwise} \end{cases}$$

From the disjointness condition $\mathbf{V}(\sigma) \cap \mathbf{V}(\Theta) \subseteq \mathbf{V}(s,t)$ we can immediately deduce
$\varphi^*|_{\mathbf{V}(s,t,\sigma)} = \pi$ [**]

Step 1

First we prove that $\lambda\tau \approx \varphi^*\sigma\ [VCOD(\tau)]$.
Let $x \in VCOD(\tau)$

Case 1 $\quad x \in \mathbf{V}(s,t)$
$\Rightarrow\ \varphi^*\sigma(x) = \pi\sigma(x)$ (see **)
$\Rightarrow\ \lambda\tau(x) \approx \pi\sigma(x) = \varphi^*\sigma(x)$ (see *)

Case 2 $\quad x \notin \mathbf{V}(s,t)$
$\Rightarrow\ x \notin DOM(\sigma)$ ($x \in VCOD(\tau)$ and $\mathbf{V}(\sigma) \cap \mathbf{V}(\Theta) \subseteq \mathbf{V}(s,t)$)
$\Rightarrow\ \varphi^*\sigma(x) = \varphi^*(x) = \lambda\tau(x)$ (Definition of φ^*) □

End of step 1

Step 2

Now we prove that $\varphi^*\sigma(\mathrm{VARS}(\tau)) \approx \varphi^*\sigma(\mathrm{TERMS}(\tau))$.

Let $x \in \mathrm{DOM}(\tau)$ with $\tau(x) = q \in \mathrm{COD}(\tau)$

$$\begin{aligned}
&\Rightarrow x \in \mathbf{V}(s,t) && (\tau \in \mu U\Sigma(s,t) \text{ and } 3.18)\\
&\Rightarrow \varphi^*\sigma(x) = \pi\sigma(x) && (\text{see } **)\\
&\qquad \approx \lambda\tau(x) && (\text{see } *)\\
&\qquad = \lambda\tau\tau(x) && (\tau \in \mathbf{\Sigma}^*)\\
&\qquad \approx \varphi^*\sigma\tau(x) && (\text{step 1 and lemma 3.5})\\
&\qquad = \varphi^*\sigma(q) &&\\
&\Rightarrow \varphi^*\sigma(\mathrm{VARS}(\tau)) \approx \varphi^*\sigma(\mathrm{TERMS}(\tau)) && \square
\end{aligned}$$

End of Step 2

With this step it is shown that φ^* unifies $\sigma(\mathrm{VARS}(\tau))$ and $\sigma(\mathrm{TERMS}(\tau))$. Next we have to prove that $\varphi^*|_W$ is equivalent to a most general element of $\mu U\Sigma(\sigma,\Theta)|_W$.

Step 3

We have proved that $\exists\, \tau \in \Theta$ with $\varphi^* \in U\Sigma(\sigma(\mathrm{VARS}(\tau)))\,,\sigma(\mathrm{TERMS}(\tau)))$.

By the definition of $\mu U\Sigma(\sigma,\Theta)|_W$ there exists a $\tau' \in \mu U\Sigma(s,t)$ and a $\varphi \in \mu U\Sigma(\sigma,\Theta)|_W$ and a $\varphi' \in U\Sigma(\sigma(\mathrm{VARS}(\tau'))\,,\sigma(\mathrm{TERMS}(\tau'))$ with $\varphi = \varphi'|_W$ and $\varphi^* \le \varphi'\ [W]$.

$$\begin{aligned}
&\text{With } \varphi'\sigma(\mathrm{VARS}(\tau)) \approx \varphi'\sigma(\mathrm{TERMS}(\tau)) &&\\
&\qquad \Rightarrow \varphi'\sigma s \approx \varphi'\sigma t && (\text{Lemma } 3.21)\\
&\qquad \Rightarrow \varphi\sigma s \approx \varphi\sigma t &&\\
&\qquad \Rightarrow \varphi \in U\Sigma(\sigma s,\sigma t) &&\\
&\text{With } \varphi^* \le \varphi'\ [W] &&\\
&\qquad \Rightarrow \pi \le \varphi\ [W] && (\varphi^*|_W = \pi)\\
&\qquad \Rightarrow \pi \equiv \varphi\ [W] && (\pi \in \mu U\Sigma(\sigma s,\sigma t) \text{ and } 3.20)
\end{aligned}$$

and that completes the proof of i). □

Proof ii)

Let $\varphi \in \mu U\Sigma(\sigma,\Theta)|_W$. We look for a $\pi \in \mu U\Sigma(\sigma s,\sigma t)$ with $\varphi \equiv \pi\ [W]$.

By the definition of $\mu U\Sigma(\sigma,\Theta)|_W$ there exists a $\tau \in \mu U\Sigma(s,t)$ and a $\varphi^* \in U\Sigma(\sigma(\mathrm{VARS}(\tau))\,,\sigma(\mathrm{TERMS}(\tau))$ with $\varphi = \varphi^*|_W \in \mu U\Sigma(\sigma,\Theta)|_W$.

$$\begin{aligned}
&\Rightarrow \varphi^*\sigma s \approx \varphi^*\sigma t && (\text{Lemma } 3.21)\\
&\Rightarrow \varphi\sigma s \approx \varphi\sigma t &&\\
&\Rightarrow \exists\, \pi \in \mu U\Sigma(\sigma s,\sigma t) \text{ with } \varphi \le \pi\ [W] &&\\
&\Rightarrow \exists\, \varphi' \in \mu U\Sigma(\sigma,\Theta)|_W \text{ with } \varphi' \equiv \pi\ [W] && (\text{Proof i}))\\
&\Rightarrow \varphi \le \varphi'\ [W] &&\\
&\Rightarrow \varphi = \varphi' && (\text{Def. } 3.17,\text{b})\\
&\Rightarrow \varphi \equiv \pi\ [W] &&
\end{aligned}$$

End of Proof ii)

From proof i) and proof ii) and lemma 3.16 we can conclude the theorem:

$$\mu U\Sigma(\sigma s,\sigma t) \equiv \mu U\Sigma(\sigma,\Theta)|_W\ [W]$$

Corollary

For two terms or literals s,t and a substitution σ: $\mu U\Sigma(s,t) = \emptyset \Rightarrow \mu U\Sigma(\sigma s,\sigma t) = \emptyset$.

5. Abstract Clause Graphs

The original formulation of the connection graph calculus [Ko75] had only one type of links: resolution links connecting Robinson unifiable literals with opposite signs in different clauses. This "pure" connection graph calculus was later augmented by other link types supporting different graph operations like subsumption recognition [Ei81], tautology and purity checking [Wa82], paramodulation [SW80] etc. Except for paramodulation links which are links between terms, all these links are links between literals and have to be treated very similar in an actual implementation. Therefore it is advantageous to describe all these different literal link types within one single calculus.

5.1 Definition (Abstract Clause Graph)

Let $\approx \subseteq \mathbf{WST} \times \mathbf{WST}$ be a stable congruence relation (definition 3.1) and $\{\approx_i \mid i = 1, \dots, n\}$ be a set of extensions of the $\approx$-relation to literals (definition 3.13) with an associated unification algorithm for each $\approx_i$. Let **D** be a set of variable disjoint clauses.

A four tupel $G(\mathbf{D}) := (\text{LNODES}, \psi, \sim, \text{LINKS})$ is called an ***abstract clause graph*** over **D** iff:

a) LNODES is an arbitrary set. (The set of literal nodes.)

b) ψ is a mapping ψ: LNODES → Literals(**D**) from the set LNODES to the literals occurring in the clauses **D**, i.e. each node is labelled with a literal.
(We write $\mathcal{L}, \mathcal{K}$ etc. for literal nodes and $\psi(\mathcal{L}) = L$ for literals)

c) $\sim$ is an equivalence relation on LNODES:
$\forall \mathcal{L}, \mathcal{K} \in \text{LNODES}\ \ \mathcal{L} \sim \mathcal{K}$ iff $\exists D \in \mathbf{D}\ \ \psi(\mathcal{L}) = L \in D$ and $\psi(\mathcal{K}) = K \in D$.
(The equivalence classes of $\sim$ just represent the clauses of **D**.)

d) $\text{LINKS} = \{(\approx_i, X_i, XIW_i, XI_i) \mid i = 1, \dots, n\}$ with

$$X_i = \{(\mathcal{L},\mathcal{K},\Theta) \mid \mathcal{L},\mathcal{K} \in \text{LNODES}, \mathcal{L} \nsim \mathcal{K},\ \emptyset \neq \Theta \subseteq \mu U\Sigma_i(L, K)\ \}$$

$$XIW_i = \{(\mathcal{L},\mathcal{K},\varrho,\Theta) \mid \mathcal{L},\mathcal{K} \in \text{LNODES}, \mathcal{L} \sim \mathcal{K},\ \varrho \in REN_{\_}(\mathbf{V}(L)) \text{ with } VCOD(\varrho) \cap \mathbf{V}(K)) = \emptyset,\ \emptyset \neq \Theta \subseteq \mu U\Sigma_i(\varrho L, K)\}$$

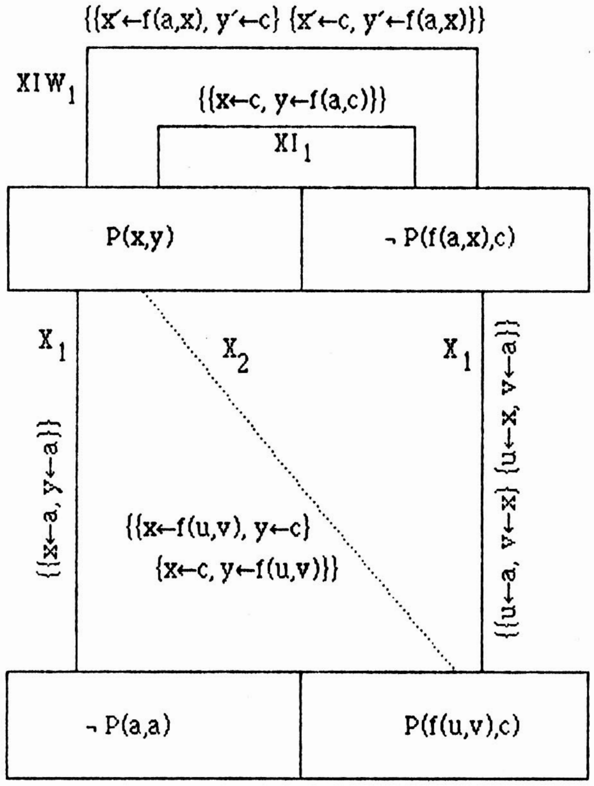
$$XI_i = \{(\mathcal{L},\mathcal{K},\Theta) \mid \mathcal{L},\mathcal{K} \in \text{LNODES}, \mathcal{L} \sim \mathcal{K},\ \emptyset \neq \Theta \subseteq \mu U\Sigma_i(L, K))\ \}$$

X_i- Links connect $\approx_i$-unifiable literals in different clauses;
XIW_i-Links connect weakly $\approx_i$-unifiable (unifiable after renaming of one literal) literals in the same clause and
XI_i-Links connect $\approx_i$-unifiable links in the same clause.

An Example

Let $\approx$ be the relation between terms as defined by the commutativity of the function f. Let $\approx_1$ be the relation between literals with opposite sign whose corresponding terms are in the $\approx$ relation. In addition assume that the symbol P denotes a symmetric predicate. Let $\approx_2$ be the relation connecting literals with equal sign and predicate, whose corresponding terms are in the $\approx$-relation. For example:
$g(f(a,b),c) \approx g(f(b,a),c)$, $P(f(a,b),c) \approx_1 \neg P(c,f(b,a))$ and $P(f(a,b),c) \approx_2 P(c,f(b,a))$.
(In the clause graph terminology [KMR83] $\approx_1$ stands for the R-Link family and $\approx_2$ for the S-Link family. $X_{1,2}$ are the R and S Links, $XIW_{1,2}$ are the RIW and SIW Links and $XI_{1,2}$ are the RI and SI Links.)

A graphical representation for a clause graph may look like:



Remark

The X-Link and XI-Link partial graph is always undirected because of the symmetry of the $\approx_i$-relations. The XIW-links however are directed because it is the <u>first</u> literal which is renamed in $\mu U\Sigma_i(\varrho L, K)$.

5.2 Definition (total abstract clause graph)

For $i \in \{1, \ldots, n\}$ an abstract clause graph $G(\mathbf{D}) := (\mathrm{LNODES}, \psi, \sim, \mathrm{LINKS})$ is

a) X_i-total iff all possible X_i-links exist and are attached with all possible mgus, i.e $\forall \mathcal{L}, \mathcal{K} \in \mathrm{LNODES}, \mathcal{L} \approx \mathcal{K}, \Theta = \mu U\Sigma_i(L, K) \neq \emptyset \Rightarrow \exists \Theta' \equiv \Theta\, [\mathbf{V}(L, K)]$ and $(\mathcal{L}, \mathcal{K}, \Theta') \in X_i$.

b) XIW_i-total iff all possible XIW_i-links exist and are attached with all possible mgus, i.e. $\forall \mathcal{L}, \mathcal{K} \in \mathrm{LNODES}, L \sim \mathcal{K}, \Theta = \mu U\Sigma_i(\varrho L, K) \neq \emptyset$ for some $\varrho \in \mathrm{REN}_{-}(\mathbf{V}(L, K))$ $\Rightarrow \exists \Theta' \equiv \Theta\, [\mathbf{V}(\varrho L, K)]$ and $(\mathcal{L}, \mathcal{K}, \varrho, \Theta') \in XIW_i$ or $(\mathcal{K}, \mathcal{L}, \varrho, \Theta') \in XIW_i$.

c) XI_i-total iff all possible XI_i-links exist and are attached with all possible mgus, i.e. $\forall \mathcal{L}, \mathcal{K} \in \mathrm{LNODES}, \mathcal{L} \sim \mathcal{K}, \Theta = \mu U\Sigma_i(L, K) \neq \emptyset \Rightarrow \exists \Theta' \equiv \Theta\, [\mathbf{V}(L, K)]$ and $(\mathcal{L}, \mathcal{K}, \Theta') \in XI_i$.

G is ***total*** iff for all $i = 1, \ldots, n$ G is X_i-total, XIW_i-total and XI_i-total.

Finally we have to define operations like resolution, factorization etc. upon abstract clause graphs. Fortunately it is possible to combine a whole class of operations and describe them with one mechanism. This class covers all the operations which take a number of literals from the graph, instantiate them with one common substitution and put them as a new clause into the graph.

5.3 Definition (Ω-Operations)

Let $G(\mathbf{D}) := (\text{LNODES}, \psi, \sim, \text{LINKS})$ be an abstract clause graph.
An Operation $\Omega(G, \mathcal{C}, \varrho, \sigma)$ is called an Ω-***Operation*** with substitutions $\sigma \in \mathbf{WFE}^*$ and $\varrho \in \text{REN}\_(\mathbf{V}(G))$ with $\text{COD}(\varrho) \cap \mathbf{V}(G) = \emptyset$ iff

a) $G' = \Omega(G, \mathcal{C}, \varrho, \sigma) =: (\text{LNODES} \cup \mathcal{C}, \psi', \sim', \text{LINKS}')$ is an abstract clause graph.
($\mathcal{C}$ represents a new clause).

b) each new literal node has exactly one parent literal node:
$\forall \mathcal{L} \in \mathcal{C}\ \exists_1 \mathcal{K} \in \text{LNODES}$ with $\psi'(\mathcal{L}) = \varrho\sigma\, \psi(\mathcal{K})$
(we use $\mathcal{K} = \mathcal{L}^\dagger$ resp. $K = L^\dagger = \psi'(\mathcal{L}^\dagger)$ as an abbreviation for this relation.
$\mathcal{L}^\dagger$ is the "parent literal node" and $L^\dagger$ the "parent literal" of $\mathcal{L}$.)
and $\psi'|_{\text{LNODES}} = \psi$
and $\sim'|_G = \sim$ and $\mathcal{C}$ is a new equivalence class of the $\sim'$ relation

c) If $\text{LINKS} = \{(\approx_i, X_i, XIW_i, XI_i) \mid i = 1, \dots, n\}$
then $\text{LINKS}' = \{(\approx_i, X_i \cup XN_i, XIW_i \cup XIWN_i, XI_i \cup XIN_i) \mid i = 1, \dots, n\}$
and each new link is connected to at least one element of $\mathcal{C}$.

XN_i, $XIWN_i$ and XIN_i are the links connecting the new clause with the old graph. Optimized formulas for the computation of the new links from the old ones are given in chapter 7.

Typical Ω-Operations are resolution, factoring, and UR-Resolution. Hyperresolution and E-resolution are Ω-Operations as long as only one copy of each clause involved is used. If more than one copy is necessary, these clauses have to be copied explicitely. Paramodulation, too, does not fit completely into this definition because the paramodulated literal itself is not an instance of an already existing literal. The other literals in a paramodulant however can be treated with this mechanism.
Also "self resolution", (resolution with two copies of the same clause) is not an Ω-Operation because two new literals may have the same parent literal.
For example: $\{\neg \underbrace{Px, Pf}(x), Qx\} \Rightarrow \{\neg Px', Pf(f(x')), Qx', Qf(x')\}$

6. Special Inheritance Mechanisms

This chapter establishes some special inheritance mechanisms for links in abstract clause graphs. The formulas for calculating the new unifiers from the old ones in special situations in the graph are derived from theorem 4.1.
The pictures at the beginning of each paragraph illustrates a situation typical for the inheritance of links to a resolvent. Boxes represent literals and a string of boxes represents a clause.

The following situations are possible and are described in detail in the next paragraphs:

An X-Link connecting a new and an old literal can be inherited from:
an X-Link connecting

- one parent literal and an unconcerned literal	(Paragraph 6.1.i) or
- two parent literals	(Paragraph 6.1.ii) or
an XIW-Link connecting two parent literals	(Paragraph 6.4).

An XIW-Link can be inherited from

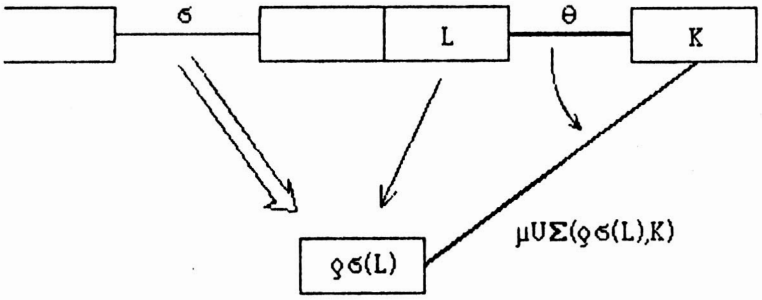
- an X-Link connecting two parent literals	(Paragraph 6.3) or
- an XIW-Link connecting two parent literals	(Paragraph 6.6).

An XI-Link can be inherited from

- an X-Link connecting two parent literals	(Paragraph 6.2) or
- an XIW-Link connecting two parent literals	(Paragraph 6.5).

6.1 X-Links → X-Links

i) Non-Parallel Case: (Connection of the new clause with the unconcerned clauses.)



6.1 Theorem

Let

a) $\Theta = \mu U\Sigma(L, K)$

b) $\sigma \in \mathbf{WS\Sigma}^*$ with $\sigma K = K$

c) $\varrho \in REN_{\subseteq}(G)$ with $\varrho K = K$ and $COD(\varrho) \cap \mathbf{V}(L, K, \Theta, \sigma) = \emptyset$ and $\mathbf{V}(\sigma L) \subseteq DOM(\varrho)$

d) $W := \mathbf{V}(\varrho\sigma L, K)$ and $W' := \mathbf{V}(\sigma L, K) - \mathbf{V}(\sigma L, \sigma K)$

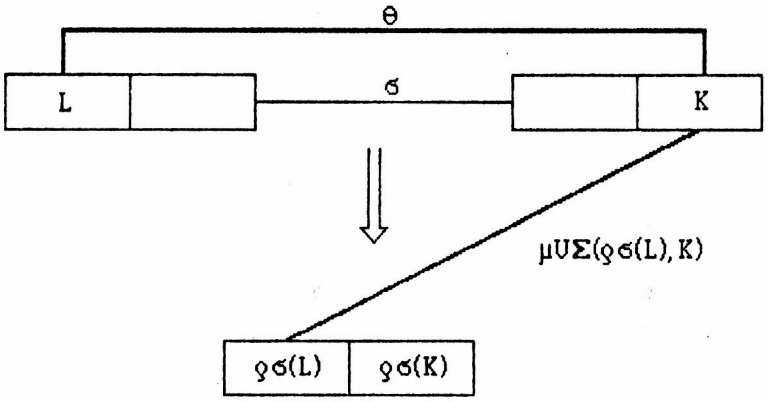
then $\mu U\Sigma(\varrho\sigma L, K) \equiv \varrho \bullet \mu U\Sigma(\sigma, \Theta)|_{W'}$ $[W]$

Proof

$$\begin{array}{ll}
\mu U\Sigma(\varrho\sigma L, K) & \\
\quad = \mu U\Sigma(\varrho\sigma L, \varrho\sigma K) & \text{(b and c)} \\
\quad = \varrho\bullet\ \mu U\Sigma(\sigma L, \sigma K) & \text{(Lemma 3.19)} \\
\quad \equiv \varrho\bullet\ \mu U\Sigma(\sigma, \Theta)|_{W'}\ [W'] & \text{(Theorem 4.1, 3.12)}
\end{array}$$

■

ii) <u>General Case</u> (Connection of the new clause with the parent clauses)



6.2 Theorem

Let

a) $\mathbf{V}(L) \cap \mathbf{V}(K) = \emptyset$

b) $\Theta = \mu U\Sigma(L, K)$

c) $\sigma \in \mathbf{WS\Sigma}^*$

d) $\varrho \in REN_{\subseteq}(G)$ with and $COD(\varrho) \cap \mathbf{V}(L, K, \theta, \sigma) = \emptyset$ and $\mathbf{V}(\sigma L, \sigma K) \subseteq DOM(\varrho)$

e) $W := \mathbf{V}(\varrho\sigma L, K)$

then $\mu U\Sigma(\varrho\sigma L, K) \equiv \mu U\Sigma((\varrho\bullet\sigma)\varrho|_{\mathbf{V}(L)}, \Theta)|_W$ [W]

Proof

$\mu U\Sigma(\varrho\sigma L, K)$

$= \mu U\Sigma((\varrho\bullet\sigma)\varrho L, K)$ (Lemma 2.14)

$= \mu U\Sigma((\varrho\bullet\sigma)\varrho|_{\mathbf{V}(L)}L, \varrho|_{\mathbf{V}(L)}K)$ (a)

$= \mu U\Sigma((\varrho\bullet\sigma)\varrho|_{\mathbf{V}(L)}L, (\varrho\bullet\sigma)\varrho|_{\mathbf{V}(L)}K)$ ($DOM(\varrho\bullet\sigma) \cap V(K) = \emptyset$)

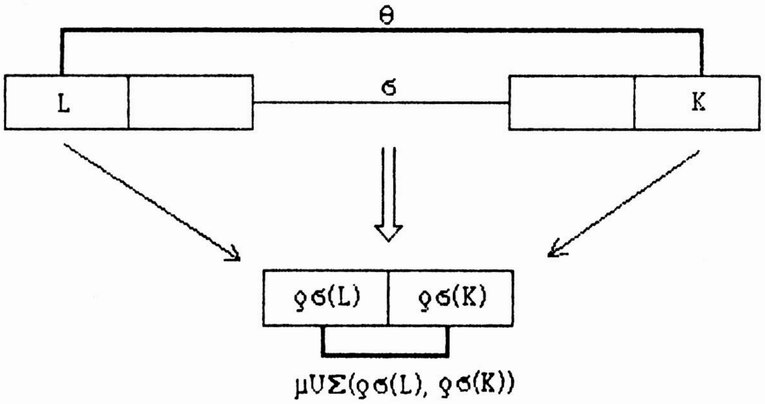
$\approx \mu U\Sigma((\varrho\bullet\sigma)\varrho|_{\mathbf{V}(L)}, \Theta)|_W$ [W] (Theorem 4.1)

■

Corollary $\mu U\Sigma(\varrho\sigma(L), K) \neq \emptyset \Rightarrow \mu U\Sigma(L, K) \neq \emptyset$ ■

6.2 X-Links → XI-Links

A new XI-link can be inherited from an old X-link connecting the parent clauses.



6.2 Theorem

Let

a) $\mathbf{V}(L) \cap \mathbf{V}(K) = \emptyset$

b) $\Theta = \mu U\Sigma(L, K)$

c) $\sigma \in \mathbf{WS\Sigma}^*$

d) $\varrho \in \text{REN}_{\subseteq}(G)$ with and $\text{COD}(\varrho) \cap \mathbf{V}(L, K, \theta, \sigma) = \emptyset$ and $\mathbf{V}(\sigma L, \sigma K) \subseteq \text{DOM}(\varrho)$

e) $W := \mathbf{V}(\varrho\sigma L, \varrho\sigma K)$ and $W' := \mathbf{V}(\sigma L, \sigma K)$

then $\quad \mu U\Sigma(\varrho\sigma L, \varrho\sigma K) \equiv \varrho \bullet \mu U\Sigma(\sigma, \Theta)|_{W'} \; [W]$

Proof

$\mu U\Sigma(\varrho\sigma L, \varrho\sigma K)$

$= \varrho \bullet \mu U\Sigma(\sigma L, \sigma K)$ (Lemma 3.19)

$\equiv \varrho \bullet \mu U\Sigma(\sigma, \Theta)|_{W'} \; [W]$ (Theorem 4.1, 3.12)

■

Corollary

$\mu U\Sigma(\varrho\sigma L, \varrho\sigma K) \neq \emptyset \Rightarrow \mu U\Sigma(L, K) \neq \emptyset$ ■

6.4 Theorem

The preconditions are the same as for theorem 6.3. Then

$\mu U\Sigma(\varrho\sigma L, \varrho\sigma K) \neq \emptyset \;\Rightarrow\;$ a) $\mu U\Sigma(\varrho\sigma L, K) \neq \emptyset$ and
b) $\mu U\Sigma(L, \varrho\sigma K) \neq \emptyset$.

Proof by contradiction

a) Let $\mu U\Sigma(\varrho\sigma L, K) = \emptyset$

$\Rightarrow \mu U\Sigma(\varrho\sigma\ \varrho\sigma L, \varrho\sigma K) = \mu U\Sigma(\varrho\sigma L, \varrho\sigma K) = \emptyset$

$(\text{VCOD}(\varrho) \cap \text{DOM}(\sigma) = \emptyset \Rightarrow \varrho\sigma \in \Sigma^*)$

$\Rightarrow$ Contradiction!

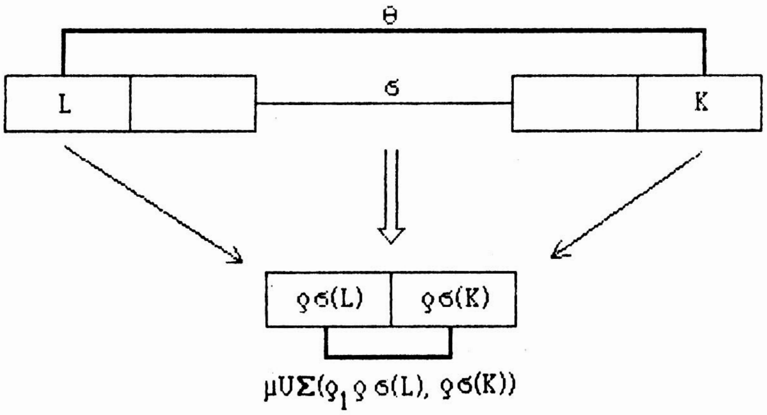
b) analogous ■

The consequence of this theorem is:

Inheritance of X-links to XI-links is only possible if both X-links between ϱσL and K as well as between L and ϱσK can be inherited too.

6.5 X-Links → XIW-Links

A new XIW-link can be inherited from an old X-link connecting the parent clauses.



6.5 Theorem

Let

a) $\mathbf{V}(L) \cap \mathbf{V}(K) = \emptyset$

b) $\Theta = \mu U\Sigma(L, K)$

c) $\sigma \in \mathbf{WS\Sigma}^*$

d) $\varrho \in REN_{\subseteq}(G)$ with and $COD(\varrho) \cap \mathbf{V}(L, K, \Theta, \sigma) = \emptyset$ and $\mathbf{V}(\sigma L, \sigma K) \subseteq DOM(\varrho)$

e) $\varrho_1 \in REN_{=}(\varrho\sigma L\ , \varrho\sigma K)$ with $COD(\varrho_1) \cap \mathbf{V}(L, K, \Theta, \sigma, \varrho) = \emptyset$

f) $W := \mathbf{V}(\varrho_1 \varrho\sigma L, \varrho\sigma K)$

then

$$\mu U\Sigma(\varrho_1 \varrho\sigma L, \varrho\sigma K) \equiv \mu U\Sigma(\varrho\sigma(\varrho_1 \varrho\sigma)|_{\mathbf{V}(L)}, \Theta)|_W\ [W]$$

Proof

1. $(\varrho_1\varrho\sigma)|_{\mathbf{V}(L)}(K) = K$ (a)

2. $(\varrho\sigma)\ (\varrho_1\varrho\sigma)|_{V(L)}(L) = (\varrho_1\varrho\sigma)|_{\mathbf{V}(L)}(L)$

(because $DOM(\varrho_1) \supseteq \mathbf{V}(\varrho\sigma L)$ and $COD(\varrho_1) \cap \mathbf{V}(\sigma) = \emptyset$)

$\mu U\Sigma(\varrho_1\ \varrho\sigma L, \varrho\sigma K)$

$= \mu U\Sigma((\varrho\sigma)(\varrho_1\varrho\sigma)|_{\mathbf{V}(L)}(L)\ , (\varrho\sigma)(\varrho_1\varrho\sigma)|_{\mathbf{V}(L)}(K))$ (1 , 2)

$\equiv \mu U\Sigma((\varrho\sigma)(\ \varrho_1\varrho\sigma)|_{\mathbf{V}(L)}\ , \Theta)|_W\ [W]$ (Theorem 4.1)

■

Corollary

$\mu U\Sigma(\varrho_1\varrho\sigma L, \varrho\sigma K) \neq \emptyset \rightarrow \mu U\Sigma(L, K) \neq \emptyset$ and

$\mu U\Sigma(\varrho\sigma L, \varrho_1\varrho\sigma K) \neq \emptyset \Rightarrow \mu U\Sigma(L, K) \neq \emptyset$ ■

6.6 Theorem

The preconditions are the same as in theorem 6.5

$$\mu U\Sigma(\varrho_1\varrho\sigma L,\ \varrho\sigma K) \neq \emptyset \Rightarrow \text{a) } \mu U\Sigma(\varrho\sigma L,\ K) \neq \emptyset \quad \text{and}$$
$$\text{b) } \mu U\Sigma(L,\ \varrho\sigma K) \neq \emptyset$$

Proof by Contradiction

a) Let $\mu U\Sigma(\varrho\sigma L,\ K) = \emptyset$

$\Rightarrow \mu U\Sigma(\varrho_1\varrho\sigma L,\ \varrho_1 K) = \emptyset$ (Corollary 4.1)

$\Rightarrow \mu U\Sigma(\varrho_1\varrho\sigma L,\ K) = \emptyset$ $(DOM(\varrho_1) \cap \mathbf{V}(K)) = \emptyset)$

$\Rightarrow \mu U\Sigma(\varrho\sigma\ \varrho_1\varrho\sigma L,\ \varrho\sigma K) = \emptyset$ (Corollary 4.1)

$\Rightarrow \mu U\Sigma(\varrho_1\varrho\sigma L,\ \varrho\sigma K) = \emptyset$

$(\varrho\sigma\ \varrho_1\varrho\sigma L = \varrho_1\varrho\sigma L$ because $DOM(\varrho_1) \subseteq \mathbf{V}(\varrho\sigma L)$ and
$DOM(\varrho\sigma) \cap VCOD(\varrho_1) = \emptyset)$

$\Rightarrow$ Contradiction! □

b) Let $\mu U\Sigma(L,\ \varrho\sigma K) = \emptyset$

$\Rightarrow \mu U\Sigma(\varrho\sigma L,\ \varrho_1\varrho\sigma K) = \emptyset$ (see a))

$\Rightarrow \varrho_1 \bullet \mu U\Sigma(\varrho\sigma L,\ \varrho_1\varrho\sigma K) = \emptyset$

$\Rightarrow \mu U\Sigma(\hat{\varrho}_1\varrho\sigma L,\ \hat{\varrho}_1\varrho_1\varrho\sigma K) = \emptyset$ (Definition 3.16 d)

$\Rightarrow \mu U\Sigma(\varrho_1\varrho\sigma L,\ \varrho\sigma K) = \emptyset$ (Lemma 2.9)

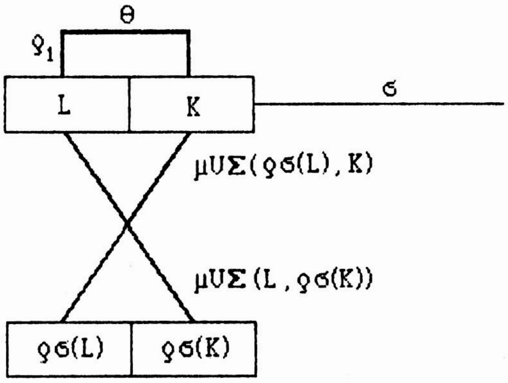
$\Rightarrow$ Contradiction! ■

The consequence of this theorem is:

Inheritance of X-Links to XIW-Links is only possible if both X-Links between $\varrho\sigma L$ and K as well as between L and $\varrho\sigma K$ can be inherited too (see also Theorem 6.4).

6.4 XIW-Links → X-Links

New X-links connecting a new clause with one of its own parent clauses can be inherited from old XIW-links.


θ
ϱ_1
L
K
σ
$\mu U\Sigma(\varrho\sigma(L), K)$
$\mu U\Sigma\ (L\ ,\ \varrho\sigma(K))$
$\varrho\sigma(L)$
$\varrho\sigma(K)$

6.7 Theorem

Let

a) $\varrho_1 \in \mathrm{REN}_=(L, K)$ with $\mathrm{VCOD}(\varrho_1) \cap V(L, K, \sigma) = \emptyset$

b) $\Theta = \mu U\Sigma(\varrho_1 L, K)$

c) $\sigma \in \mathbf{WS\Sigma}^*$

d) $\varrho \in \mathrm{REN}_\subseteq(G)$ with $\mathrm{VCOD}(\varrho) \cap \mathbf{V}(L, K, \Theta, \sigma, \varrho_1) = \emptyset$ and $\mathbf{V}(\sigma L, \sigma K) \subseteq \mathrm{DOM}(\varrho)$

e) $\varrho_2 \in \mathrm{REN}_=(\mathrm{VCOD}(\varrho_1))$ with $\varrho \approx \varrho_2 \varrho_1 [\mathrm{DOM}(\varrho_1)]$

f) $V := \mathbf{V}(\varrho\sigma L, K)$ and $W := V(\varrho L, \sigma K)$

then

$$\mu U\Sigma(\varrho\sigma L, K) \equiv \mu U\Sigma((\varrho \bullet \sigma)\, \varrho_2, \Theta)|_V\ [V]$$

Proof.

1. $\varrho_2 K = K$ (because $\mathrm{DOM}(\varrho_2) = \mathrm{VCOD}(\varrho_1)$ and a) $\Rightarrow \mathrm{DOM}(\varrho_2) \cap \mathbf{V}(K) = \emptyset$)

2. $\mu U\Sigma(\varrho\sigma L, K)$
   - $= \mu U\Sigma((\varrho \bullet \sigma)\, \varrho L, K)$ (Lemma 2.14)
   - $= \mu U\Sigma((\varrho \bullet \sigma)\, \varrho_2 \varrho_1 L, K)$ (e)
   - $= \mu U\Sigma((\varrho \bullet \sigma)\, \varrho_2 \varrho_1 L, (\varrho \bullet \sigma)\, \varrho_2 K)$ (a, d, e, 1)
   - $\equiv \mu U\Sigma((\varrho \bullet \sigma)\, \varrho_2, \Theta)|_V\ [V]$ (Theorem 4.1)

■

6.8 Theorem

The preconditions are the same as in theorem 6.7. Then

$$\mu U\Sigma(L, \varrho\sigma K) \equiv \varrho \bullet \mu U\Sigma(\sigma \varrho_2, \Theta)|_W\ [\mathbf{V}(L, \varrho\sigma K)]$$

Proof

$\mu U\Sigma(L, \varrho\sigma K)$
- $= \mu U\Sigma(L, \varrho\sigma\, \varrho_2 K)$ ($\varrho_2 K = K$)
- $= \mu U\Sigma(\hat{\phi}\, \varrho L, \varrho\sigma\, \varrho_2 K)$ (Lemma 2.9)
- $= \mu U\Sigma(\hat{\phi}\, \varrho L, \hat{\phi}\, \sigma\, \varrho_2 K)$ (Lemma 2.9)
- $= \varrho \bullet \mu U\Sigma(\varrho L, \sigma\, \varrho_2 K)$ (Lemma 3.19)
- $= \varrho \bullet \mu U\Sigma(\sigma\, \varrho L, \sigma\, \varrho_2 K)$ ($\mathrm{DOM}(\sigma) \cap \mathbf{V}(\varrho L) = \emptyset$)
- $= \varrho \bullet \mu U\Sigma(\sigma\, \varrho_2 \varrho_1 L, \sigma\, \varrho_2 K)$ (e)
- $\equiv \varrho \bullet \mu U\Sigma(\sigma\, \varrho_2, \Theta)|_W\ [\mathbf{V}(L, \varrho\sigma K)]$ (Theorem 4.1)

■

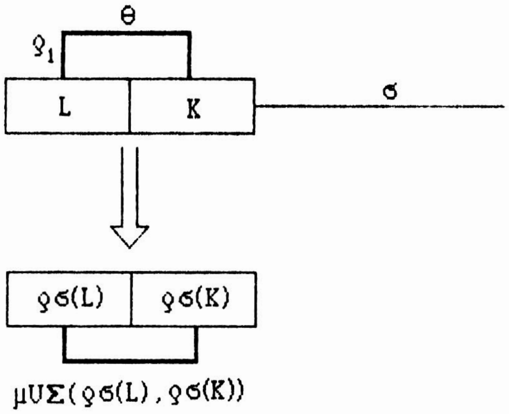
Corollary

$\mu U\Sigma(\varrho\sigma L, K) \neq \emptyset \Rightarrow \mu U\Sigma(\varrho_1 L, K) \neq \emptyset$ and

$\mu U\Sigma(L, \varrho\sigma K) \neq \emptyset \Rightarrow \mu U\Sigma(\varrho_1 L, K) \neq \emptyset$ ■

6.5 XIW-Links → XI-Links

New XI-links can be inherited from old XIW-links.



6.9 Theorem

Let

a) $\varrho_1 \in REN_=(L, K)$ with $VCOD(\varrho_1) \cap \mathbf{V}(L, K, \sigma) = \emptyset$

b) $\Theta = \mu U\Sigma(\varrho_1 L, K)$

c) $\sigma \in \mathbf{WS\Sigma}^*$

d) $\varrho \in REN_\subseteq(G)$ with $VCOD(\varrho) \cap \mathbf{V}(L, K, \Theta, \sigma, \varrho_1) = \emptyset$ and $\mathbf{V}(\sigma L, \sigma K) \subseteq DOM(\varrho)$

e) $\varrho_2 \in REN_=(VCOD(\varrho_1))$ with $\varrho \approx \varrho_2 \varrho_1 [DOM(\varrho_1)]$

f) $W := V(\varrho\sigma L, \varrho\sigma K)$

then

$$\mu U\Sigma(\varrho\sigma L, \varrho\sigma K) \equiv \mu U\Sigma((\varrho\bullet\sigma)\ \varrho\varrho_2, \Theta\rangle_W\ [W]$$

Proof

$\mu U\Sigma(\varrho\sigma L, \varrho\sigma K)$

$= \mu U\Sigma((\varrho\bullet\sigma)\ \varrho L, (\varrho\bullet\sigma)\varrho K)$ (Lemma 2.14)

$= \mu U\Sigma((\varrho\bullet\sigma)\ \varrho\varrho L, (\varrho\bullet\sigma)\varrho K)$ ($\varrho \in \mathbf{\Sigma}^*$)

$= \mu U\Sigma((\varrho\bullet\sigma)\ \varrho\ \varrho_2 \varrho_1 L, (\varrho\bullet\sigma)\varrho K)$ (c)

$= \mu U\Sigma((\varrho\bullet\sigma)\ \varrho\ \varrho_2 \varrho_1 L, (\varrho\bullet\sigma)\varrho\ \varrho_2 K)$ ($DOM(\varrho_2) \cap \mathbf{V}(K) = \emptyset$)

$\equiv \mu U\Sigma((\varrho\bullet\sigma)\ \varrho\varrho_2, \Theta\rangle_W\ [W]$ (Theorem 4.1)

■

Corollary

$$\mu U\Sigma(\varrho\sigma L, \varrho\sigma K) \neq \emptyset \Rightarrow \mu U\Sigma(\varrho_1 L, K) \neq \emptyset$$ ■

6.10 Theorem

$$\mu U\Sigma(\varrho\sigma L, \varrho\sigma K) \neq \emptyset \Rightarrow \text{a) } \mu U\Sigma(\varrho\sigma L, K) \neq \emptyset \text{ and } \text{b) } \mu U\Sigma(L, \varrho\sigma K) \neq \emptyset$$

Proof by Contradiction

a) Let $\mu U\Sigma(\varrho\sigma L, K) = \emptyset$

$\Rightarrow \mu U\Sigma(\varrho\sigma\ \varrho\sigma\ L, \varrho\sigma\ K) = \emptyset$ (Corollary 4.1)

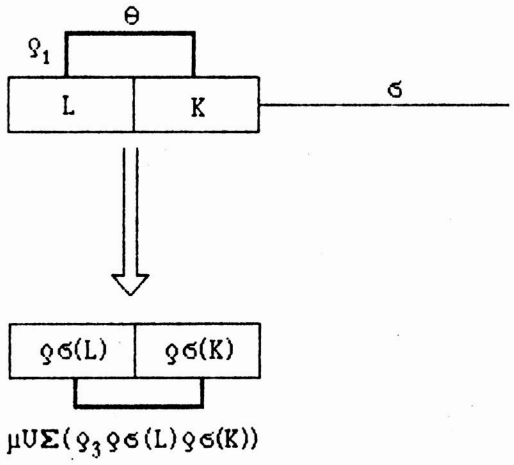
$\Rightarrow \mu U\Sigma(\varrho\sigma\ L, \varrho\sigma\ K) = \emptyset$ (Idempotence of $\varrho\sigma$)

⇒ Contradiction!

b) analogous. ■

6.6 XIW-Links → XIW-Links

New XIW-links can be inherited from old XIW-links.



6.11 Theorem

Let

a) $\varrho_1 \in \mathrm{REN}_=(L, K)$ with $\mathrm{VCOD}(\varrho_1) \cap \mathbf{V}(L, K, \sigma) = \emptyset$

b) $\Theta = \mu U\Sigma(\varrho_1 L, K)$

c) $\sigma \in \mathbf{WS\Sigma}^*$

d) $\varrho \in \mathrm{REN}_{\sqsubseteq}(G)$ with $\mathrm{VCOD}(\varrho) \cap V(L, K, \Theta, \sigma, \varrho_1) = \emptyset$ and

$\mathbf{V}(\sigma L, \sigma K) \subseteq \mathrm{DOM}(\varrho)$

e) $\varrho_2 \in \mathrm{REN}_=(\mathrm{VCOD}(\varrho_1))$ with $\varrho \approx \varrho_2 \varrho_1 [\mathrm{DOM}(\varrho_1)]$

f) $\varrho_3 \in \mathrm{REN}_=(\varrho\sigma L, \varrho\sigma K)$ with $\mathrm{VCOD}(\varrho_3) \cap \mathbf{V}(L, K, \Theta, \sigma, \varrho_1, \varrho) = \emptyset$

g) $\mathbf{W} := \mathbf{V}(\varrho_3 \varrho\sigma L, \varrho\sigma K)$ and $\mathbf{W}' = \mathbf{V}(\varrho\sigma L, \sigma K)$

then

$$\mu U\Sigma(\varrho_3 \varrho\sigma L, \varrho\sigma K) \equiv (\varrho\, \varrho_3) \bullet \mu U\Sigma(\sigma\, (\varrho\bullet\sigma)\, \varrho_2, \Theta)|_{W'} [W]$$

Proof

$\mu U\Sigma(\varrho_3 \varrho\sigma L, \varrho\sigma K)$	
$= \mu U\Sigma(\varrho_3 \sigma\varrho\sigma L, \varrho\sigma K)$	(d)
$= \mu U\Sigma(\varrho\varrho_3 \sigma\varrho\sigma L, \varrho\sigma K)$	$(\mathrm{DOM}(\varrho) \cap \mathbf{V}(\varrho_3 \varrho\sigma L) = \emptyset$
$= \mu U\Sigma(\varrho\varrho_3 \sigma(\varrho\bullet\sigma)\, \varrho L, \varrho\sigma K)$	(Lemma 2.14)
$= \mu U\Sigma(\varrho\varrho_3 \sigma(\varrho\bullet\sigma)\, \varrho_2 \varrho_1 L, \varrho\sigma K)$	(e)
$= \mu U\Sigma(\varrho\varrho_3 \sigma(\varrho\bullet\sigma)\, \varrho_2 \varrho_1 L, \varrho\varrho_3 \sigma K)$	$(\mathrm{DOM}(\varrho_3) \cap \mathbf{V}(\sigma K) = \emptyset$
$= \mu U\Sigma(\varrho\varrho_3 \sigma(\varrho\bullet\sigma)\, \varrho_2 \varrho_1 L, \varrho\varrho_3 \sigma \varrho_2 K)$	$(\varrho_2(K) = K)$
$= (\varrho\varrho_3) \bullet \mu U\Sigma(\sigma(\varrho\bullet\sigma)\, \varrho_2 \varrho_1 L, \sigma \varrho_2 K)$	
$= (\varrho\varrho_3) \bullet \mu U\Sigma(\sigma(\varrho\bullet\sigma)\, \varrho_2 \varrho_1 L, \sigma\, (\varrho\bullet\sigma)\, \varrho_2 K)$	$(\mathrm{DOM}(\varrho\bullet\sigma) \cap \mathbf{V}(\varrho_2 K) = \emptyset$
$\equiv (\varrho\varrho_3) \bullet \mu U\Sigma\, (\sigma(\varrho\bullet\sigma)\, \varrho_2, \Theta)|_{W'} [W]$	■

Corollary

$\mu U\Sigma(\varrho_3 \varrho\sigma L, \varrho\sigma K) \neq \emptyset \Rightarrow \mu U\Sigma(\varrho_1 L, K) \neq \emptyset$ and

$\mu U\Sigma(\varrho\sigma L, \varrho_3 \varrho\sigma K) \neq \emptyset \Rightarrow \mu U\Sigma(\varrho_1 L, K) \neq \emptyset$ ■

6.12 Theorem

$$\mu U\Sigma(\varrho_3\varrho\sigma L,\ \varrho\sigma K) \neq \emptyset \Rightarrow \text{a) } \mu U\Sigma(\varrho\sigma L,\ K) \neq \emptyset \text{ and}$$
$$\text{b) } \mu U\Sigma(L,\ \varrho\sigma K) \neq \emptyset$$

Proof by Contradiction

a) Let $\mu U\Sigma(\varrho\sigma L,\ K) = \emptyset$

$\Rightarrow \mu U\Sigma(\varrho_3\varrho\sigma L,\ \varrho_3 K) = \emptyset$ (Corollary 4.1)

$\Rightarrow \mu U\Sigma(\varrho_3\varrho\sigma L,\ K) = \emptyset$ ($\varrho_3 K = K$)

$\Rightarrow \mu U\Sigma(\varrho\sigma\ \varrho_3\varrho\sigma L,\ \varrho\sigma K) = \emptyset$ (Corollary 4.1)

$\Rightarrow \mu U\Sigma(\varrho_3\varrho\sigma L,\ \varrho\sigma K) = \emptyset$

($\varrho\sigma\ \varrho_3\varrho\sigma L = \varrho_3\varrho\sigma L$ because $DOM(\varrho_3) \subseteq \mathbf{V}(\varrho\sigma L)$ and $DOM(\varrho\sigma) \cap VCOD(\varrho_3) = \emptyset$)

$\Rightarrow$ Contradiction! □

b) Let $\mu U\Sigma(L,\ \varrho\sigma K) = \emptyset$

$\Rightarrow \mu U\Sigma(\varrho\sigma L,\ \varrho_3\varrho\sigma K) = \emptyset$ (see a))

$\Rightarrow \varrho_3 \bullet \mu U\Sigma(\varrho\sigma L,\ \varrho_3\varrho\sigma K) = \emptyset$

$\Rightarrow \mu U\Sigma(\hat{\varrho}_3\ \varrho\sigma L,\ \hat{\varrho}_3\ \varrho_3\varrho\sigma K) = \emptyset$ (Lemma 3.19)

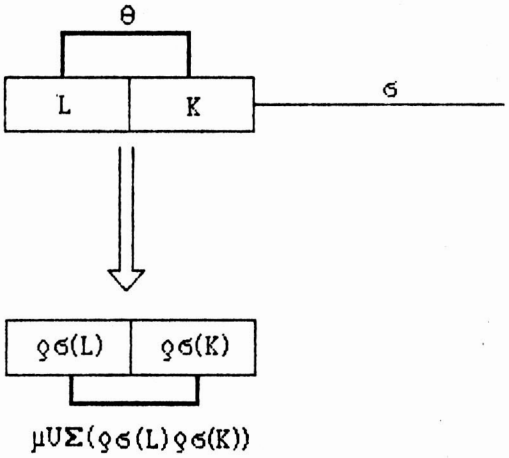
$\Rightarrow \mu U\Sigma(\varrho_3\ \varrho\sigma L,\ \varrho\sigma K) = \emptyset$ (Lemma 2.9)

$\Rightarrow$ Contradiction! ■

(see also Theorem 6.6)

6.7 XI-Links → XI-Links

New XI-links can be inherited from old XI-links.



6.13 Theorem

Let

a) $\Theta = \mu U\Sigma(L,\ K)$

b) $\sigma \in \mathbf{WS\Sigma}^*$

c) $\varrho \in REN_{\sqsubseteq}(G)$ with $VCOD(\varrho) \cap \mathbf{V}(L, K, \Theta, \sigma, \varrho_1) = \emptyset$ and $\mathbf{V}(\varrho\sigma L,\ \varrho\sigma K) \subseteq DOM(\varrho)$

d) $W := \mathbf{V}(\varrho\sigma L,\ \varrho\sigma K)$ and $W' := \mathbf{V}(\sigma L,\ \sigma K)$

then

$$\mu U\Sigma(\varrho\sigma L,\ \varrho\sigma K) \equiv \varrho \bullet \mu U\Sigma(\sigma\ ,\Theta)|_{W'}\ [W]$$

The proof is obvious.

6.14 Theorem

Let L and K be literals and $\varrho \in \mathrm{REN}_{-}(L)$, then $\mu U\Sigma(L, K) \neq \emptyset \Rightarrow \mu U\Sigma(\varrho L, K) \neq \emptyset$

Proof by Contradiction

Let $\mu U\Sigma(\varrho L, K) = \emptyset$

$\Rightarrow \mu U\Sigma(\varrho\varrho L, \varrho K) = \emptyset$ (Corollary 4.1)

$\Rightarrow \mu U\Sigma(\varrho L, \varrho K) = \emptyset$ $(\varrho \in \mathbf{\Sigma}^*)$

$\Rightarrow \varrho \bullet \mu U\Sigma(L, K) = \emptyset$ (Lemma 3.19)

$\Rightarrow \mu U\Sigma(L, K) = \emptyset$ $\Rightarrow$ Contradiction! ■

This theorem states the fact that whenever an XI-Link is possible, there is also an XIW-Link possible. Therefore it is not necessary to inherit XI-Links, because all the new links can be inherited from the parallel XIW-Links using the theorems of the last paragraph. Sometimes, however, it is useful to inherit XI-Links directly from XI-Links in order to inherit also certain properties attached to the unifiers. Inheritance of XI-Links to X-Links and XIW-Links, however, is incomplete.

7. The General Inheritance Algorithm

The Ω-Operations, as defined in chapter 5, map clause graphs onto clause graphs taking a set of literals of the graph, applying a substitution σ and a renaming substitution ϱ to the literals and inserting them as a new clause into the graph. The remaining problem is: how to get the new links connecting the new clause with the rest of the graph without to much search. How to optimize this search in standard connection graphs is already known since 1975 [Ko75], [Br75] and there is not much difference to the inheritance algorithm in abstract clause graphs. The idea is to scour the links connected to the parent clauses and to unify the substitutions attached to these links with σ as it is described in the last chapter in order to get the unifiers for the new links. We will describe the method in this chapter and prove that it transforms total graphs into total graphs.

7.1 Theorem

Let

a) $G = (\mathrm{LNODES}, \psi, \sim, \{(\approx_i, X_i, XIW_i, XI_i) \mid i = 1, \ldots, n\}$ be a total abstract clause graph and.

b) Let Ω be an Ω-Operation and $G' = \Omega(G, \mathcal{C}, \varrho, \sigma)$ be an abstract clause graph generated by Ω. $G' = (\mathrm{LNODES} \cup \mathcal{C}\ \psi', \sim', \{(\approx_i, X_i \cup XN_i, XIW_i \cup XIWN_i, XI_i \cup XIN_i) \mid i = 1, \ldots, n\}$

If we calculate the new links according to the following formulas:

c) $XN_i = \{(\mathcal{L}, \mathcal{K}, \mu U\Sigma_i(L, K)) \mid \mathcal{L} \in \mathcal{C} \wedge \mathcal{K} \notin \mathcal{C}, \mu U\Sigma_i(L, K) \neq \emptyset$ and

$[(\mathcal{L}', \mathcal{K}, \mu U\Sigma_i(L', K)) \in X_i$ or (see 6.1)

$(\mathcal{L}', \mathcal{K}, \varrho_1, \mu U\Sigma_i(\varrho_1 L', K)) \in XIW_i$ or (see 6.4)

$(\mathcal{K}, \mathcal{L}', \varrho_1, \mu U\Sigma_i(L', \varrho_1 K)) \in XIW_i\,]\}$ (see 6.4)

d) $XIWN_i = \{(\mathcal{L}, \mathcal{K}, \varrho_3, \mu U\Sigma_i(\varrho_3 L, K)) \mid \mathcal{L} \in \mathcal{C} \wedge \mathcal{K} \in \mathcal{C}, \mu U\Sigma_i(\varrho_3 L, K) \neq \emptyset$ and

$[(\mathcal{L}', \mathcal{K}', \mu U\Sigma_i(L', K')) \in X_i$ or (see 6.3)

$(\mathcal{L}', \mathcal{K}', \varrho_1, \mu U\Sigma_i(\varrho_1 L', K')) \in XIW_i$ or (see 6.6)

$(\mathcal{K}', \mathcal{L}', \varrho_1, \mu U\Sigma_i(L', \varrho_1 K')) \in XIW_i]\}$ (see 6.6)

e) $XIN_i = \{(\mathcal{L}, \mathcal{K}, \mu U\Sigma_i(L, K)) \mid \mathcal{L} \in \mathcal{C} \wedge \mathcal{K} \in \mathcal{C}, \mu U\Sigma_i(L, K) \neq \emptyset$ and

$[(\mathcal{L}', \mathcal{K}', \mu U\Sigma_i(L', K')) \in X_i$ or (see 6.2)

$(\mathcal{L}', \mathcal{K}', \varrho_1, \mu U\Sigma_i(\varrho_1 L', K')) \in XIW_i$ or (see 6.5)

$(\mathcal{K}', \mathcal{L}', \varrho_1, \mu U\Sigma_i(L', \varrho_1 K')) \in XIW_i\,]\}$

then G′ is again a total graph.

Proof

Let $i \in \{1, \dots, n\}$

i) Let $\mathcal{L}, \mathcal{K} \in \text{LNODES} \cup \mathcal{C}$, $\mathcal{L} \nsim \mathcal{K}$, $\Theta = \mu U\Sigma(L, K) \neq \emptyset$

We have to prove that $(\mathcal{L}, \mathcal{K}, \Theta) \in X_i \cup XN_i$.

Case 1 $\mathcal{L} \notin \mathcal{C}$ and $\mathcal{K} \notin \mathcal{C}$

$\Rightarrow (\mathcal{L}, \mathcal{K}, \Theta) \in X_i$ because G is total.

Case 2 w.L.o.G $\mathcal{K} \notin \mathcal{C}$ and $\mathcal{L} \in \mathcal{C}$ with $L = \varrho\sigma L'$

Case 2.1 $\mathcal{L}' \nsim \mathcal{K}$

$\Rightarrow \mu U\Sigma_i(L', K) \neq \emptyset$ (Corollary 6.2)

$\Rightarrow (\mathcal{L}', \mathcal{K}, \mu U\Sigma_i(L', K)) \in X_i$

$\Rightarrow (\mathcal{L}, \mathcal{K}, \Theta) \in XN_i$

Case 2.2 $\mathcal{L}' \sim \mathcal{K}$

$\Rightarrow \mu U\Sigma_i(\varrho_1 L', K) \neq \emptyset$ (Corollary 6.8)

$\Rightarrow (\mathcal{L}', \mathcal{K}, \mu U\Sigma_i(L', K)) \in X_i$ or $(\mathcal{K}, \mathcal{L}', \mu U\Sigma_i(L', \varrho_1 K)) \in X_i$

$\Rightarrow (\mathcal{L}, \mathcal{K}, \Theta) \in XN_i$

$\Rightarrow (\mathcal{L}, \mathcal{K}, \Theta) \in X_i \cup XN_i$.

ii) Let $\mathcal{L}, \mathcal{K} \in \text{LNODES} \cup \mathcal{C}$, $\mathcal{L} \sim \mathcal{K}$, $\Theta = \mu U\Sigma_i(\varrho_3 L, K) \neq \emptyset$

We have to prove that $(\mathcal{L}, \mathcal{K}, \varrho_3, \Theta) \in XIW_i \cup XIWN_i$ or

$(\mathcal{K}, \mathcal{L}, \varrho_3, \mu U\Sigma_i(L, \varrho_3 K)) \in XIW_i \cup XIWN_i$.

Case 1 $\mathcal{L} \notin \mathcal{C}, \mathcal{K} \notin \mathcal{C}$

$\Rightarrow (\mathcal{L}, \mathcal{K}, \varrho_3, \Theta) \in XIW_i$ or $(\mathcal{K}, \mathcal{L}, \varrho_3, \mu U\Sigma_i(L, \varrho_3 K)) \in XIW_i$

(because G is total.)

Case 2 $\mathcal{L} \in \mathcal{C}, \mathcal{K} \in \mathcal{C}$

Case 2.1 $L' \nsim K'$

$\Rightarrow \mu U\Sigma_i(L', K') \neq \emptyset$ (Corollary 6.5)

$\Rightarrow (\mathcal{L}', \mathcal{K}', \mu U\Sigma_i(L', K') \in X_i$

$\Rightarrow (\mathcal{L}, \mathcal{K}, \varrho_3, \Theta) \in XIWN_i$

Case 2.2 $L' \sim K'$

$\Rightarrow \mu U\Sigma_i(\varrho_1 L', K') \neq \emptyset$ (Corollary 6.11)

$\Rightarrow (\mathcal{L}', \mathcal{K}', \varrho_1, \mu U\Sigma_i(\varrho_1 L', K')) \in XIW_i$ or $(\mathcal{K}', \mathcal{L}', \varrho_1, \mu U\Sigma_i(L', \varrho_1 K' \in XIW_i$

$\Rightarrow (\mathcal{L}, \mathcal{K}, \varrho_3, \Theta) \in XIWN_i$ or $(\mathcal{K}, \mathcal{L}, \varrho_3, \mu U\Sigma_i(L, \varrho_3 K)) \in XIWN_i$.

We have $\forall \mathcal{L}, \mathcal{K} \in \mathcal{C}$ $\mathcal{L} \sim \mathcal{K}$, therefore $\mathcal{L} \in \mathcal{C}$ and $\mathcal{K} \notin \mathcal{C}$ is not possible.

$\Rightarrow (\mathcal{L}, \mathcal{K}, \varrho_3, \Theta) \in XIWN_i$ or $(\mathcal{K}, \mathcal{L}, \varrho_3, \mu U\Sigma_i(L, \varrho_3 K)) \in XIWN_i$.

iii) Let $\mathcal{L}, \mathcal{K} \in \text{LNODES} \cup \mathcal{C}$, $\mathcal{L} \sim \mathcal{K}$, $\Theta = \mu U\Sigma_i(L, K) \neq \emptyset$

$\Rightarrow \mu U\Sigma_i(\varrho_3 L, K) \neq \emptyset$ for an appropriate ϱ_3. (Theorem 6.14)

The proof for $(\mathcal{L}, \mathcal{K}, \Theta) \in XI_i \cup XIN_i$ is similar to ii)

i), ii), iii) and Definition 5.2 $\Rightarrow$ G′ is total. ■

Further Remarks

The algorithm can be optimized in the following way: Each X-Link and XIW-Link can be inherited under certain conditions to two X-Links, one XIW-Link and one XI-Link. Exploiting theorems 6.4, 6.6 and 6.12 one can organize the algorithm in a way that whenever two X-Links can potentially be inherited, but the unifiers for at least one of them are empty, no attempt to inherit an XIW-Link and an XI-Link need be made because their unifiers are empty too. Furthermore, if the two X-Links can be inherited, but the unifiers for the XIW-Link are empty then the unifiers for the XI-Link will be empty too and need not be computed.

8. Summary

Two goals have been achieved with this work:

1. The variety of different types of links which have been used so far in the clause graph calculus has been described within a unique framework:
   Three link types are combined into one link family, the common characteristic of which is a relation between literals. This relation has to be defined for each link family such that a unifier can be calculated, the application of which makes the relation hold between two literals. The unification algorithms on the literal level are supported by a common theory-unification algorithm for the unification of two terms according to a given congruence relation for terms. The three link types in a link familiy only differ in that one type connects unifiable literals in different clauses (X-Links), whereas the other two types connect unifiable literals in the same clause (XI-Links) resp. weakly unifiable (unifiable after renaming of one literal) literals in the same clause (XIW-Links).

2. It is possible to define the link inheritance mechanism for a relatively large class of operations on the clause graph (resolution, factoring, hyperresolution, E-resolution etc.) completely independently of the respective literal relation. The only necessary rules are those for inheriting the three link types inside a link family. The literal relations themselves need only be used during the construction of the graph and may be ignored afterwards. In order to achieve this independence, our analysis has shown how to calculate the mgu-set for instances of two literals directly from the mgu-set of the two literals without unifying these instances again. This method was demonstrated for a very general class of many sorted calculi and congruence relations on terms. The extensions of the term relation to literals have also been kept very general, since only certain compatibility conditions are required.

The advantage of the new inheritance mechanism is not only that it is usually (but not always!) cheaper to calculate the new unifiers from the old ones, but that an ancestor relation between the new and old unifiers can be established which allows the inheritance of properties attached to the unifiers to their descendants and to use these properties for new deletion rules in the clause graph calculus.

Acknowledgements
I would like to thank my colleagues N. Eisinger, A. Herold, M. Schmidt-Schauss, H.J. Bürckert and Ch. Lingenfelder for their support during the preparation of this work. J. Siekmann read two drafts of this paper. His helpful critisism and support contributed very much to its present form.

References

[Br75] Bruynooghe, M. 'The Inheritance of Links in a Connection Graph'. Report CW2 (1975). Applied Mathematics and Programming Division. Katholieke Universiteit Leuven.

[Ei81] Eisinger, N. 'Subsumption and Connection Graphs'. Proc. of IJCAI-81, Vancouver (1981)

[He83] Herold, A. 'Some Basic Notions of First-Order Unification Theory'. Interner Bericht 15/83, Inst. für Informatik I, Univ. of Karlsruhe, (1983).

[KMR83] Karl Mark G. Raph, 'The Markgraf Karl Refutation Procedure'. Interner Bericht, Memo-Seki-MK-84-01, FB Informatik, Univ. of Kaiserslautern (1984).

[Ko75] Kowalski, R. 'A Proof Procedure Using Connection Graphs'. J.ACM 22,4, (1975).

[Si84] Siekmann, J. 'Universal Unification' Proc. of CADE-84, Nappa USA. Springer (1984).

[SS85] Schmidt-Schauss, M. 'A Many-Sorted Calculus with Polymorphic Functions Based on Resolution and Paramodulation'. Proc. of IJCAI-85, Los Angeles (1985).

[St83] Stickel, M.E. 'Theory Resolution: Building in Non-Equational Theories'. SRI Report, (1983).

[SW80] Siekmann, J., Wrightson, G. 'Paramodulated Connection Graphs'. Acta Informatica (1978).

[Sz82] Szabo, P. 'Unifikationstheorie erster Ordnung' Dissertation, Inst. für Informatik I, Univ. of Karlsruhe (1982).

[Wa81] Walther, Ch. 'Elimination of Redundant Links in Extended Connection graphs'. Proc. of GWAI-81, Springer Fachberichte (1981) and Interner Bericht 10/81, University of Karlsruhe.

[Wa82] Walther, Ch. 'A Many-Sorted Calculus Based on Resolution and Paramodulation'. Interner Bericht 34/82 Inst. für Informatik I, Univ. of Karlsruhe (1982). see also Proc. of IJCAI-83, Karlsruhe (1983).

Index of the Symbols Used

Signature (Chapter 2)

V	set of all variable symbols	**F**	set of all function symbols
P	set of all predicate symbols	**T**	set of all terms
L	set of all literals	**V**(O)	set of all variables occurring in O

Substitutions (Chapter 2)

DOM(σ)	domain of σ
COD(σ)	codomain of σ = σ(DOM(σ))
VARS(σ)	domain of σ regarded as a list.
TERMS(σ)	σ(VARS(σ)) ≈ COD(σ)
VCOD(σ)	variables in the codomain of σ
$\mathbf{\Sigma}$	set of all substitutions
$\mathbf{\Sigma}^*$	set of idempotent substitutions
REN$_=$(V)	renaming substitutions ϱ with DOM(ϱ) = V
REN$_\subseteq$(V)	renaming substitutions ϱ with DOM(ϱ) ⊆ V
ϱ^c	converse of the renaming substitution ϱ
$\hat{\varrho}$	permutation of order 2, generated by ϱ
•	renaming of a substitution: $\varrho \bullet \sigma = \hat{\varrho}\sigma\hat{\varrho}$

Sorted Logic (Def. 2.2)

[...]	sort of a term
WS	a predicate selecting well sorted terms and literals
WST, **WSL**, **WSΣ**, **WSΣ***	well sorted terms, literals, substitutions and idempotent substitutions

Relations

≤	partial ordering on sorts	
≈	1. basic congruence relation on terms	(Def. 3.1)
	2. "natural" extension to literals	(Def. 3.13)
≈ [W]	strong extension of ≈ to substitutions	(Def. 3.3)
∝ [W]	canonical extension of the ≈-relation to literals	(Def 3.13)
≤ [W]	a subsumption relation on substitutions	(Def. 3.6)
≡ [W]	an equivalence relation on substitutions	(Def. 3.6)

Unifiers

UΣ(s,t)	complete set of unifiers for s and t (terms or literals).	(Def. 3.16)
μUΣ(s,t)	set of most general unifiers for s and t	
μUΣ(σ,Θ)$|_W$	set of most general "unifiers" for the substitution σ and the set of substitutions Θ, restricted to the variables W.	(Chapter 4)
MAX$|_W$(Θ)	An instances removing function for sets of substitutions.	(Def. 3.17)

Clause Graphs (Chapter 5)

LNODES set of literal nodes in the graph.
(Multiple occurrences of literals are possible in a clause graph. Therefore the literal nodes in a clause graph are different from the corresponding literals. The literal nodes are written with letters like $\mathcal{L}$ or $\mathcal{K}$ and the corresponding literal is then L resp. K).

~ A relation on literal nodes grouping them into "clauses".

Ω-Operation
An operation on clause graphs, taking a set of literals, instantiating them, grouping them into a new clause, and inserting them into the graph.