CLASSICAL LOGICS FOR ATTRIBUTE-VALUE LANGUAGES

Jürgen Wedekind
Xerox Palo Alto Research Center
and
C.S.L.I. - Stanford University
USA

Abstract

This paper describes a classical logic for attribute-value (or feature description) languages which are used in unification grammar to describe a certain kind of linguistic object commonly called attribute-value structure (or feature structure). The algorithm which is used for deciding satisfiability of a feature description is based on a restricted deductive closure construction for sets of literals (atomic formulas and negated atomic formulas). In contrast to the Kasper/Rounds approach (cf. [Kasper/Rounds 90]), we can handle cyclicity, without the need for the introduction of complexity norms, as in [Johnson 88] and [Beierle/Pletat 88]. The deductive closure construction is the direct proof-theoretic correlate of the congruence closure algorithm (cf. [Nelson/Oppen 80]), if it were used in attribute-value languages for testing satisfiability of finite sets of literals.

1 Introduction

This paper describes a classical logic for attribute-value (or feature description) languages which are used in unification grammar to describe a certain kind of linguistic object commonly called attribute-value structure (or feature structure). From a logical point of view an attribute-value structure like e.g. the following (in matrix notation)

$$a\left[\begin{array}{ll} \text{PRED} & \text{'PROMISE'} \\ \text{TENSE} & \text{PAST} \\ \text{SUBJ} & \boxed{1}\left[\begin{array}{ll}\text{PRED} & \text{'JOHN'}\end{array}\right] \\ \text{XCOMP} & \left[\begin{array}{ll}\text{SUBJ} & \boxed{1} \\ \text{PRED} & \text{'COME'}\end{array}\right] \end{array}\right]$$

can be regarded as a graphical representation of a minimal model of a satisfiable feature description. If we assume that the attributes (in the example: PRED, TENSE, SUBJ, XCOMP) are unary partial function symbols and the values (a, 'PROMISE', PAST, 'JOHN', 'COME') are constants then the given feature structure represents graphically e.g. the minimal model of the following description:

PRED SUBJ$a \approx$ 'JOHN' & TENSE$a \approx$ PAST &
PRED$a \approx$ 'PROMISE' & SUBJ$a \approx$ SUBJ XCOMPa &
PRED XCOMP$a \approx$ 'COME'.[1]

So, in the following attribute-value languages are regarded as quantifier-free sublanguages of classical first order languages with equality whose (nonlogical) symbols are given by a set of unary partial function symbols (attributes) and a set of constants (atomic and complex values). The logical vocabulary includes all propositional connectives; negation is interpreted classically.[2]

For quantifier-free attribute-value languages L we give an axiomatic or Hilbert type system H^0_{AV} which simply results from an ordinary first order system (with partial function symbols), if its language were restricted to the vocabulary of L. According to requirements of the applications, axioms for the constant-consistency, constant/complex-consistency and acyclicity can be added to force these properties for the feature structures (models).

For deciding consistency (or satisfiability) of a feature description, we assume first, that the conjunction of the formulas in the feature description is converted to disjunctive normal form. Since a formula in disjunctive normal form is consistent, iff at least one of its disjuncts is consistent, we only need an algorithm for deciding consistency of finite sets of literals (atomic formulas or negated atomic formulas) S. In contrast to the reduction algorithms which normalize a set S according to a complexity norm in a sequence of norm decreasing rewrite steps[3] we use a restricted deductive closure algorithm for deciding the consistency of sets of literals.[4] The restriction results from the fact that it is sufficient for deciding the consistency of S to consider proofs of equations from S with a certain subterm property. For the closure construction only those equations are derived from S whose terms are subterms of the terms occurring in the formulas of S. This guarantees that the construction terminates with a finite set of literals. The adequacy of this subterm property restriction, which was already shown for the number theoretic calculus K in [Kreisel/Tait 61] by [Statman 74], is a necessary condition for the development of more efficient Cut-free Gentzen type systems for attribute-

[1] Note that the terms are formed without using brackets. (Since all function symbols are unary, the introduction of brackets would not improve the readability essentially.) Therefore we write e.g. PRED SUBJa instead of PRED(SUBJ(a)).

[2] For intuitionistic negation cf. e.g. [Dawar/Vijay-Shanker 90] and [Langholm 89].

[3] Cf. e.g. [Kreisel/Tait 61], [Knuth/Bendix 70], and applied to attribute-value languages [Johnson 88], [Beierle/Pletat 88], [Smolka 89].

[4] Since we allow cyclicity, unrestricted deductive closure algorithms (cf. e.g. [Kasper/Rounds 86] and [Kasper/Rounds 90]) cannot be applied.

value languages.[5]

Moreover, this closure construction is the direct proof-theoretic correlate of the congruence closure algorithm (cf. [Nelson/Oppen 80]), if it were used for testing satisfiability of finite sets of literals in H^0_{AV}. As it is shown there, the congruence closure algorithm can be used to test consistency if the terms of the equations are represented as labeled graphs and the equations as a relation on the nodes of that graph.

On the basis of the algorithm for deciding satisfiability of finite sets of formulas we then show the completeness and decidability of H^0_{AV}.

2 Attribute-Value Languages

In this section we define the type of language we want to consider and introduce some additional notation.

2.1 Syntax

2.1. DEFINITION. A quantifier-free attribute-value language ($L\epsilon\mathcal{L}^0_{AV}$) consists of the logical connectives $\bot$ (false), $\sim$ (negation), $\supset$ (implication), the equality symbol $\approx$ and the parentheses (,). The *nonlogical* vocabulary is given by a finite set of *constants* C and a finite set of *unary partial function symbols* F_1 ($C \cap F_1 = \emptyset$).

2.2. DEFINITION. The class of *terms* ($\mathcal{T}$) of L is recursively defined as follows: each constant is a term; if f is a function symbol and τ is a term, then $f\tau$ is a term.

2.3. DEFINITION. The set of *atomic* formulas of L is $\{\tau_1 \approx \tau_2 \mid \tau_1, \tau_2 \epsilon \mathcal{T}\} \cup \{\bot\}$.

2.4. DEFINITION. The *formulas* of L are the atomic formulas and, whenever ϕ and ψ are formulas, then so are $(\sim \phi)$ and $(\phi \supset \psi)$.

2.5. DEFINITION. If α is a well-formed expression (term or formula), then $\alpha[\tau_1/\tau_2]$ is used to designate an expression obtained from α by replacing *some* (possibly all or none) occurrences of τ_1 in α by τ_2.

We assume that the connectives $\vee$ (disjunction), & (conjunction) and $\equiv$ (equivalence) are introduced by their usual definitions. Furthermore, we write sometimes $\tau_1 \not\approx \tau_2$ instead of $\sim \tau_1 \approx \tau_2$ and drop the parentheses according to the usual conventions.[6]

2.2 Semantics

A *model* for L consists of a nonempty *universe* $\mathcal{U}$ and an *interpretation* function $\Im$. Since not every term denotes an element in $\mathcal{U}$ if the function symbols are interpreted as unary partial functions, we generalize the partiality of the denotation by assuming that $\Im$ itself is a partial function. Thus in general not all of the constants and function symbols are interpreted by $\Im$. Redundancies which result from the fact that non-interpreted function symbols and function symbols interpreted as empty functions are then regarded as distinct are removed by requiring these partial functions to be nonempty. Suppose $[X \mapsto Y]_{(p)}$ designates the set of all (partial) functions from X to Y, then a model is defined as follows:

2.6. DEFINITION. A *model* for L is a pair $M = \langle \mathcal{U}, \Im \rangle$, consisting of a nonempty set $\mathcal{U}$ and an interpretation function $\Im = \Im_C \cup \Im_{F_1}$, such that

(i) $\Im_C \epsilon [C \mapsto \mathcal{U}]_p$
(ii) $\Im_{F_1} \epsilon [F_1 \mapsto [\mathcal{U} \mapsto \mathcal{U}]_p]_p$
(iii) $\forall f \epsilon F_1 (f \epsilon \mathrm{Dom}(\Im) \rightarrow \Im(f) \neq \emptyset)$.

The (partial) *denotation* function for terms $\overline{\Im}$ ($\overline{\Im}\epsilon[\mathcal{T} \mapsto \mathcal{U}]_p$) induced by $\Im$ is defined as follows:[7]

2.7. DEFINITION. For every $c\epsilon C$ and $f\tau\epsilon\mathcal{T}$ ($f\epsilon F_1$),

$$\overline{\Im}(c) = \begin{cases} \Im(c) & \text{if } c\epsilon\mathrm{Dom}(\Im) \\ \text{undefined} & \text{otherwise} \end{cases}$$

$$\overline{\Im}(f\tau) = \begin{cases} \Im(f)(\overline{\Im}(\tau)) & \text{if } f\epsilon\mathrm{Dom}(\Im) \wedge \overline{\Im}(\tau) \text{ defined} \wedge \\ & \overline{\Im}(\tau)\epsilon\mathrm{Dom}(\Im(f)) \\ \text{undefined} & \text{otherwise.} \end{cases}$$

2.8. DEFINITION. The *satisfaction* relation between models M and formulas ϕ ($\models_M \phi$, read: M satisfies ϕ, M is a model of ϕ, ϕ is true in M) is defined recursively:

$$\begin{array}{lcl} \not\models_M \bot & & \\ \models_M \tau \approx \tau' & \leftrightarrow & \Im(\tau), \Im(\tau') \text{ defined } \wedge \Im(\tau) = \Im(\tau') \\ \models_M \sim \phi & \leftrightarrow & \neg(\models_M \phi) \\ \models_M \psi \supset \chi & \leftrightarrow & \models_M \psi \rightarrow \models_M \chi. \end{array}$$

A formula ϕ is *valid* ($\models \phi$), iff ϕ is true in all models. A formula ϕ is *satisfiable*, iff it has at least one model. Given a set of formulas Γ, we say that M satisfies Γ ($\models_M \Gamma$), iff M satisfies each formula ϕ in Γ. Γ is *satisfiable*, iff there is a model that satisfies each formula in Γ. ϕ is *logical consequence* of Γ ($\Gamma \models \phi$), iff every model that satisfies Γ is a model of ϕ.

3 The System H^0_{AV}

In this section we describe an *axiomatic* or Hilbert type system H^0_{AV} for quantifier-free attribute-value languages L. We give a decision procedure for the satisfiability of finite sets of formulas and show the completeness and decidability of H^0_{AV} on the basis of that procedure.

3.1 Axioms and Inference Rules

If L is a fixed attribute-value language, then the system consists of a traditional axiomatic propositional calculus for L and two additional equality axioms. For any formulas ϕ, ψ, χ, terms

[5] Cf. also [Statman 77].

[6] We drop the outermost brackets, assume that the connectives have the precedence $\sim > \& > \vee > \supset, \equiv$ and are left associative.

[7] In the text following the definition we drop the overline.

τ, τ', and every sequence of functors σ ($\sigma \epsilon F_1^*$) of L the formulas under A1 - A4 are *propositional* axioms[8] and the formulas under E1 and E2 are *equality* axioms.[9] The Modus Ponens (MP) is the only *inference* rule.[10]

A1 $\vdash \sim \perp$

A2 $\vdash \phi \supset (\psi \supset \phi)$

A3 $\vdash (\phi \supset (\psi \supset \chi)) \supset ((\phi \supset \psi) \supset (\phi \supset \chi))$

A4 $\vdash (\sim \phi \supset \sim \psi) \supset (\psi \supset \phi)$

E1 $\vdash \sigma\tau \approx \tau' \supset \tau \approx \tau$

E2 $\vdash \tau \approx \tau' \supset (\phi \supset \phi[\tau/\tau'])$

MP $\phi \supset \psi \wedge \phi \vdash \psi$

A formula ϕ is *derivable* from a set of formulas Γ ($\Gamma \vdash \phi$), iff there is a finite sequence of formulas $\phi_1 ... \phi_n$ such that $\phi_n = \phi$ and every ϕ_i is an axiom, one of the formulas in Γ or follows by MP from two previous formulas of the sequence. ϕ is a *theorem* ($\vdash \phi$), iff ϕ is derivable from the empty set. Δ is derivable from Γ ($\Gamma \vdash \Delta$), iff each formula of Δ is derivable from Γ. Γ and Δ are *deductively equivalent* ($\Gamma \dashv\vdash \Delta$), iff $\Gamma \vdash \Delta$ and $\Delta \vdash \Gamma$.

The system is *sound*:[11]

3.1. THEOREM. *For every formula ϕ: If $\vdash \phi$, then $\models \phi$.*

Beside this weak version also the *strong* soundness theorem is provable for H^0_{AV}:

3.2. THEOREM. *For every set of formulas Γ and every formula ϕ: If $\Gamma \vdash \phi$, then $\Gamma \models \phi$.*

3.2 Satisfiability

We now prove

3.3. THEOREM. *The satisfiability of a finite set of formulas Γ is decidable.*

by providing a terminating procedure: First the conjunction of all formulas in Γ (denoted by $\bigwedge \Gamma$) is converted into disjunctive normal form (*DNF*) using the well-known standard techniques. Then $\bigwedge \Gamma$ is equivalent with a *DNF*

$$\vdash \bigwedge\Gamma \equiv (\phi_1^1 \& \phi_2^1 \& ... \& \phi_{k_1}^1) \vee (\phi_1^2 \& ... \& \phi_{k_2}^2) \vee ... \vee (\phi_1^n \& ... \& \phi_{k_n}^n)$$

where the conjuncts ϕ_j^i ($i = 1, .., n$; $j = 1, .., k_i$) are either atomic formulas or negations of atomic formulas, henceforth called *literals*. By the definition of the satisfiability we get:

[8] Cf. e.g. [Church 56].

[9] Axiom E1 restricts the reflexivity of identity to denoting terms: if a term denotes, then also its subterms do (cf. the definition of $\Im$). Thus equality is not a reflexive, but only a subterm reflexive relation.

[10] If (i.) constant-consistency and (ii.) constant/complex-consistency are to be guaranteed for a set of atomic values V ($V \subseteq C$), for each $a, b \epsilon V$ ($a \neq b$) and $f \epsilon F_1$, axioms of the form (i.) $\vdash a \not\approx b$ and (ii.) $\vdash fa \not\approx fa$ have to be added (a finite set). If also acyclicity has to be ensured, axioms of the form (iii.) $\vdash \sigma\tau \not\approx \tau$, with $\sigma \epsilon F_1^+, \tau \epsilon T$, have to be added. Although this set is infinite, we only need a finite subset for the satisfiability test and for decidability (see below).

[11] For the propositional calculus cf. the standard proofs. For axioms E1 and E2 cf. [Johnson 88].

3.4. LEMMA. *Let $\bigwedge S^1 \vee \bigwedge S^2 \vee ... \vee \bigwedge S^n$ be a DNF of $\bigwedge \Gamma$ consisting of conjunctions $\bigwedge S^i$ of the literals in S^i, then $\bigwedge \Gamma$ is satisfiable, iff at least one disjunct $\bigwedge S^i$ is satisfiable.*

We complete the proof of Theorem 3.3 by an algorithm that converts a finite set of literals S^i into a deductively equivalent set of literals in normal form S^i_L which is satisfiable iff it is not equal to $\{\perp\}$.

3.2.1 A Normal Form for Sets of Literals

The normal form is constructed by closing S deductively by those equations whose terms are subterms of the terms occurring in S. For the construction we use the following derived rules:

R1	$\sigma\tau \approx \tau' \vdash \tau \approx \tau$	Subterm Reflexivity
R2	$\tau \approx \tau' \wedge \phi \vdash \phi[\tau/\tau']$	Substitutivity
R3	$\tau \approx \tau' \vdash \tau' \approx \tau$	Symmetry.

We get R1 and R2 from E1 and E2 by the deduction theorem. R3 is derivable from R1 and R2, since we get from $\tau \approx \tau'$ first $\tau \approx \tau$ by R1 and then $\tau' \approx \tau$ by R2.

If $\mathcal{T}_S$ denotes the set of terms occurring in the formulas of S ($\mathcal{T}_S = \{\tau, \tau' \mid (\sim)\tau \approx \tau' \epsilon S\}$), and SUB($\mathcal{T}_S$) denotes the set of all *subterms* of the terms in $\mathcal{T}_S$[12]

$$\mathrm{SUB}(\mathcal{T}_S) = \{\tau \mid \sigma\tau \epsilon \mathcal{T}_S, \text{ with } \sigma \epsilon F_1^*\},$$

then the normal form is constructed according to the following inductive definition.

3.5. DEFINITION. For a given set of literals S we define a sequence of sets S_i ($i \geq 0$) by induction:

With $S'_0 = S \cup \{\tau' \approx \tau \mid \tau \approx \tau' \epsilon S\}$,

$$S_0 = \begin{cases} \{\perp\} & \text{if } \perp \epsilon S; \text{ otherwise} \\ S'_0 \cup \{\tau \approx \tau \mid \sigma\tau \approx \tau' \epsilon S'_0\} \end{cases}$$

$$S_{i+1} = \begin{cases} \{\perp\} & \text{if } \exists \phi \epsilon S_i (\sim \phi \epsilon S_i); \text{ otherwise} \\ S_i \cup \left\{(\tau_1 \approx \tau_2)[\tau/\tau'] \middle| \begin{array}{l} \tau_1 \approx \tau_2, \tau \approx \tau' \epsilon S_i \wedge \\ \mathcal{T}_{\{(\tau_1 \approx \tau_2)[\tau/\tau']\}} \subseteq \mathrm{SUB}(\mathcal{T}_S) \end{array} \right\} \end{cases}.$$

Since $S_i \subseteq S_{i+1}$, for $S_{i+1} \neq \{\perp\}$, the construction terminates on the basis of the subterm condition either with a finite set of literals or with $\{\perp\}$. If each term of the equations in S_{i+1} is a subterm of the terms in $\mathcal{T}_S$, no term of the equations in S_{i+1} can be longer than the longest term in $\mathcal{T}_S$.

EXAMPLE 1. Assume that L consists of the constants a, b, c, e and the function symbols f, g, h, m, n, p. Then, for the set of literals

$$S = \left\{ \begin{array}{l} ge \approx pmb, e \approx me, mb \approx ngffc, c \approx a, \\ ga \approx ha, a \approx ffa, ngffa \not\approx e \end{array} \right\}$$

the following sequence of sets is constructed. We represent the equations of a set S_i by the system of sets of equivalent terms induced by S_i. I.e.: If Θ is a set of terms under S_i and

[12] $\mathcal{T}_S \subseteq$ SUB($\mathcal{T}_S$) holds by definition.

$\tau, \tau' \epsilon \Theta$, then $\tau \approx \tau' \epsilon S_i$. Furthermore, we mark by an arrow that a set under S_i is also induced (without modifications) by the equations in S_{i+1}.

S_0	S_1	$S_2 = S_\nu$
$ngffa \not\approx e$	$\rightarrow$	$\rightarrow$
$\{e, me\}$	$\rightarrow$	$\rightarrow$
$\{b\}$	$\rightarrow$	$\rightarrow$
$\{c,a\}$, $\{a,ffa\}$, $\{ffc\}$	$\{c,a,ffa,ffc\}$	$\rightarrow$
$\{ge,pmb\}$	$\rightarrow$	$\rightarrow$
$\{mb,ngffc\}$	$\{mb,ngffc,ngffa\}$	$\rightarrow$
$\{fc\}$, $\{fa\}$	$\{fc,fa\}$	$\rightarrow$
$\{gffc\}$	$\{gffc,gffa\}$	$\{gffc,gffa,ga,ha\}$
$\{ga,ha\}$	$\{ga,ha,gffa\}$	

3.6. DEFINITION. Let $S_\nu = S_t$; with $t = \min\{i \mid S_i = S_{i+1}\}$.

3.7. LEMMA. *For S_ν holds: $S \dashv\vdash S_\nu$.*

PROOF. If $S_\nu \neq \{\perp\}$, then S and S_ν are deductively equivalent, since S is a subset of S_ν and S_ν only contains formulas derivable from S. For $S_\nu = \{\perp\}$ the same holds for $S_{\nu-1}$. Since $S_{\nu-1}$ is inconsistent, S is deductively equivalent with $\{\perp\}$. □

Note that for each equation in S_i ($S_i \neq \{\perp\}$) there is a proof from S with the ***subterm property***, as defined below. This follows from the subterm condition in the inductive construction.

3.8. DEFINITION. A proof of an equation from S has the ***subterm property***, iff each term occurring in the equations of that proof is a subterm of the terms in $\mathcal{T}_S$, i.e. an element of $\text{SUB}(\mathcal{T}_S)$.

So, if S is not trivially inconsistent ($\perp$ not in S), the construction terminates with $\{\perp\}$, since there exists a proof of an equation from S with the subterm property, whose negation is in S.

EXAMPLE 2. For the inconsistent set $S' = S \cup \{gmme \not\approx pnhffa\}$ the construction terminates after 4 steps ($S'_4 = \{\perp\}$), since there is a proof of $gmme \approx pnhffa$ from S' with the subterm property of depth 3.

$$\dfrac{\dfrac{ge \approx pmb \quad \dfrac{e \approx me \quad e \approx me}{e \approx mme}}{gmme \approx pmb} \qquad \dfrac{\dfrac{mb \approx ngffc \quad c \approx a}{mb \approx ngffa} \quad \dfrac{ga \approx ha \quad a \approx ffa}{gffa \approx hffa}}{mb \approx nhffa}}{gmme \approx pnhffa}$$

The deductive closure construction restricted by the subterm property is a proof-theoretic simulation of the congruence closure algorithm (cf. [Nelson/Oppen 80][13]), if used for testing satisfiability of finite sets of literals in Π^0_{AV}. Strictly speaking, if

i. the congruence closure algorithm is weakened for partial functions,

ii. S is not trivially inconsistent ($\perp$ not in S), and

iii. the failure in the induction step of 3.5. is overruled,

then $\tau \approx \tau'$ is in S_ν iff the nodes which represent the terms τ and τ' in the graph constructed for S are congruent.[14] Moreover, for unary partial functions the algorithm is simpler, since the arity does not have to be controlled.

3.9. LEMMA. *The set of all equations in S_ν is closed under subterm reflexivity, symmetry and transitivity.*

PROOF. For $S_\nu = \{\perp\}$ trivial. If $S_\nu \neq \{\perp\}$, then S_ν is closed under subterm reflexivity and symmetry, since these properties are inherited from S_0 to its successor sets. S_ν is closed under transitivity, since we first get $\tau_3 \epsilon \text{SUB}(\mathcal{T}_S)$ from $\tau_1 \approx \tau_2, \tau_2 \approx \tau_3 \epsilon S_\nu$ and then according to the construction also $\tau_1 \approx \tau_2[\tau_2/\tau_3] \epsilon S_{\nu+1} = S_\nu$, with $\tau_2[\tau_2/\tau_3] = \tau_3$. □

3.2.2 Satisfiability of Sets of Literals

For the proof that the satisfiability of a finite set of literals is decidable we first show that a set of literals in normal form is satisfiable, iff the set is not equal to $\{\perp\}$. For $S_\nu = \{\perp\}$ we get trivially:

3.10. LEMMA. $S_\nu = \{\perp\} \rightarrow \neg\exists M(\models_M S_\nu)$.

Otherwise we can show the satisfiability of S_ν by the construction of a canonical model that satisfies S_ν.

Let E_ν be the set of all (nonnegated) equations in S_ν, $\mathcal{T}_{E_\nu}$ the set of terms occurring in E_ν and $\approx_{E_\nu}$ the relation induced by E_ν on $\mathcal{T}_{E_\nu}$ ($\{\langle\tau,\tau'\rangle \mid \tau \approx \tau' \epsilon E_\nu\}$). Then, we choose as the universe of the canonical model $M_\nu = \langle \mathcal{U}_\nu, \Im_\nu \rangle$ the set of all equivalence classes of $\approx_{E_\nu}$ on $\mathcal{T}_{E_\nu}$, if $\mathcal{T}_{E_\nu} \neq \emptyset$. By Lemma 3.9 this set exists. If S_ν contains no (unnegated) equation, we set $\mathcal{U}_\nu = \{\emptyset\}$, since the universe has to be nonempty.

3.11. DEFINITION. For a set of literals S_ν in normal form, the *canonical term model* for S_ν is given by the pair $M_\nu = \langle \mathcal{U}_\nu, \Im_\nu \rangle$, consisting of the universe

$$\mathcal{U}_\nu \;=\; \begin{cases} \mathcal{T}_{E_\nu}/\approx_{E_\nu} & \text{if } \mathcal{T}_{E_\nu} \neq \emptyset \\ \{\emptyset\} & \text{otherwise} \end{cases}$$

and the interpretation function $\Im_\nu$, which is defined for $c \epsilon C$, $f \epsilon F_1$ and $[\tau] \epsilon \mathcal{U}_\nu$ by:[15]

$$\Im_C(c) \;=\; \begin{cases} [c] & \text{if } c \epsilon \mathcal{T}_{E_\nu} \\ \text{undefined} & \text{otherwise} \end{cases}$$

$$\Im_{F_1}(f)([\tau]) \;\doteq\; \begin{cases} [f\tau'] & \text{if } \tau' \epsilon [\tau] \text{ and } f\tau' \epsilon \mathcal{T}_{E_\nu} \\ \text{undefined} & \text{otherwise.} \end{cases}$$

It follows from the definition that $\Im_\nu$ is a partial function. Suppose further for $\Im_{F_1}(f)$ that $[\tau_1] = [\tau_2]$ and that $\Im_{F_1}(f)([\tau_1])$ is defined. Then

$$\Im_{F_1}(f)([\tau_1]) = \Im_{F_1}(f)([\tau_2]).$$

For this, suppose $\Im_{F_1}(f)([\tau_1]) = [f\tau']$, with $\tau' \epsilon [\tau_1]$. Since $\approx_{E_\nu}$ is an equivalence relation we get $\tau' \epsilon [\tau_2]$ and thus $\Im_{F_1}(f)([\tau_2]) = [f\tau']$.

[13] Cf. also [Gallier 87].

[14] Cf. [Wedekind 90].

[15] We drop the $\approx_{E_\nu}$-index of the equivalence classes.

EXAMPLE 3. The canonical model for S of Example 1 which is constructed using $S_2 = S_\nu$ is given by:

$$\mathcal{U}_\nu = \left\{ \begin{array}{c} \{c, mc\}, \{b\}, \{c, a, ffa, ffc\}, \\ \{ge, pmb\}, \{mb, ngffc, ngffa\}, \\ \{fc, fa\}, \{gffc, gffa, ga, ha\} \end{array} \right\}$$

$$\Im_\nu(e) = [e] \qquad \left.\begin{array}{l}\Im_\nu(c) \\ \Im_\nu(a)\end{array}\right\} = [c]$$
$$\Im_\nu(b) = [b]$$

$$\Im_\nu(f) = \left\{ \begin{array}{c} \langle [a], [fa] \rangle, \\ \langle [fa], [ffa] \rangle \end{array} \right\} \qquad \Im_\nu(m) = \left\{ \begin{array}{c} \langle [e], [me] \rangle, \\ \langle [b], [mb] \rangle \end{array} \right\}$$

$$\Im_\nu(g) = \left\{ \begin{array}{c} \langle [e], [ge] \rangle, \\ \langle [a], [ga] \rangle \end{array} \right\} \qquad \Im_\nu(h) = \left\{ \langle [a], [ha] \rangle \right\}$$

$$\Im_\nu(n) = \left\{ \langle [ga], [ngffc] \rangle \right\} \qquad \Im_\nu(p) = \left\{ \langle [mb], [pmb] \rangle \right\}.$$

For each term τ in $\mathcal{T}_{E_\nu}$ it follows from the definition of $\Im_C$ and $\Im_{F_1}$: $\Im_\nu(\tau) = [\tau]$.

By the following lemma we show in addition that the domain of $\Im_\nu$ restricted to $\mathcal{T}_{S_\nu}$ is equal to $\mathcal{T}_{E_\nu}$.

3.12. LEMMA. *For each term τ in $\mathcal{T}_{S_\nu}$: If $\Im_\nu$ is defined for τ, then $\Im_\nu(\tau) = [\tau]$, with $\tau \epsilon \mathcal{T}_{E_\nu}$.*

PROOF. (By induction on the length of τ.) Suppose first that $\Im_\nu$ is defined for τ. For every constant c it follows from the definition of $\Im_C$ that $\Im_C(c) = [c]$, with $c \epsilon \mathcal{T}_{E_\nu}$. Assume for $f\tau$ by inductive hypothesis $\Im_\nu(\tau) = [\tau]$, with $\tau \epsilon \mathcal{T}_{E_\nu}$, then it follows from the definition of $\Im_{F_1}(f)$ that $\Im_{F_1}(f)([\tau]) = [f\tau']$, with $f\tau' \epsilon \mathcal{T}_{E_\nu}$ and $\tau' \epsilon [\tau]$. Since τ' is a subterm of $f\tau'$, we first get $\tau' \epsilon \mathcal{T}_{E_\nu}$ and by Lemma 3.9 $f\tau' \approx f\tau', \tau' \approx \tau \epsilon S_\nu$. Because of $f\tau \epsilon \mathrm{SUB}(\mathcal{T}_S)$, then also $f\tau \approx f\tau \epsilon S_\nu$. So, $f\tau$ must also be in $\mathcal{T}_{E_\nu}$ and hence $\Im_{F_1}(f)([\tau]) = [f\tau]$. □

Next we show for the model M_ν:

3.13. LEMMA. $S_\nu \neq \{\bot\} \rightarrow \models_{M_\nu} S_\nu$.

PROOF. (We prove $\models_{M_\nu} \phi$, for every ϕ in S_ν by induction on the structure of ϕ.)

$\bot$ is not element of S_ν. If $\bot$ were in S_ν, we would get by the definition of S_ν $S_\nu = \{\bot\}$ which contradicts our assumption.

For $\phi = \sim \bot$, $\models_{M_\nu} \sim \bot$ holds trivially.

Suppose $\phi = \tau \approx \tau'$, then τ, τ' are in $\mathcal{T}_{E_\nu}$, $\Im_\nu$ is defined for τ and τ', and $\Im_\nu(\tau) = [\tau]$, $\Im_\nu(\tau') = [\tau']$. Because of $\tau \approx \tau' \epsilon S_\nu$, it follows that $[\tau] = [\tau']$. So $\Im_\nu(\tau) = \Im_\nu(\tau')$ and hence $\models_{M_\nu} \tau \approx \tau'$.

Assume that ϕ is $\sim (\tau \approx \tau')$. If $\tau \approx \tau'$ were satisfied by M_ν, $\Im_\nu(\tau)$ would be equal to $\Im_\nu(\tau')$. By Lemma 3.12 we would then get $\Im_\nu(\tau) = [\tau]$ and $\Im_\nu(\tau') = [\tau']$, with $\tau, \tau' \epsilon \mathcal{T}_{E_\nu}$. Since $\approx_{E_\nu}$ is an equivalence relation on $\mathcal{T}_{E_\nu}$, $\tau \approx \tau' \epsilon S_\nu$ would follow from $[\tau] = [\tau']$, and, contradicting the assumption, we would get $S_\nu = \{\bot\}$ by the definition of S_ν. □

It can be easily shown that M_ν is a unique (up to isomorphism) minimal model for S_ν.[16] Strictly speaking, if M is a model for S_ν homomorphic to M_ν, then every minimal submodel of M that satisfies S_ν is isomorphic to M_ν.

[16] It can be verified very easily by using this fact that we need to add to a set of literals S only a finite number of axioms to ensure the acyclicity. All axioms of the form $\sigma\tau \not\approx \tau$ $(\sigma \epsilon F_1^+, \tau \epsilon \mathcal{T})$, with $|\sigma\tau| \leq |\mathrm{SUB}(\mathcal{T}_E)|$, are e.g. more than enough, since from a consistent but cyclic set of literals S must follow an equation $\sigma\tau \approx \tau$ $(\sigma \epsilon F_1^+, \tau \epsilon \mathcal{T})$, with $|\sigma\tau| \leq |\mathcal{U}_\nu|$, and $|\mathcal{U}_\nu| \leq |\mathrm{SUB}(\mathcal{T}_E)|$ holds by the construction of $\mathcal{U}_\nu$.

From the two lemmata above it follows first that the satisfiability of sets of formulas in normal form is decidable:

$$S_\nu \neq \{\bot\} \leftrightarrow \exists M(\models_M S_\nu).$$

Since S_ν and S are deductively equivalent, we can establish by the following lemma that the satisfiability of arbitrary finite sets of literals S is decidable.

3.14. LEMMA. $S_\nu \neq \{\bot\} \leftrightarrow \exists M(\models_M S)$.

PROOF. ($\rightarrow$) If $S_\nu \neq \{\bot\}$, we know by Lemma 3.13 that M_ν is a model for S_ν. Then, by the soundness $S_\nu \vdash S \rightarrow \forall M(\models_M S_\nu \rightarrow \models_M S)$. Since S is derivable from S_ν, it follows $\models_{M_\nu} S$ and thus $S_\nu \neq \{\bot\} \rightarrow \exists M(\models_M S)$.

($\leftarrow$) If $S_\nu = \{\bot\}$, then for each model $M \not\models_M S_\nu$. From the soundness we get $S \vdash S_\nu \rightarrow \forall M(\models_M S \rightarrow \models_M S_\nu)$. Since S_ν is derivable from S, it follows $\forall M(\not\models_M S_\nu \rightarrow \not\models_M S)$ and hence $S_\nu = \{\bot\} \rightarrow \forall M(\not\models_M S)$. □

3.3 Completeness and Decidability

Using the procedure for deciding satisfiability we can easily show the *completeness* and *decidability* of H^0_{AV}.

3.15. THEOREM. *For every finite set of formulas Γ, and for each formula ϕ: If $\Gamma \models \phi$, then $\Gamma \vdash \phi$.*

PROOF. By definition ϕ is a logical consequence of Γ, iff $\Gamma \cup \{\sim \phi\}$ is unsatisfiable. Using the equivalences of Theorem 3.3, we first get:

$$\Gamma \cup \{\sim \phi\} \dashv\vdash \{\textstyle\bigwedge(\Gamma \cup \{\sim \phi\})\}.$$

Suppose, that $\bigwedge S^1 \vee ... \vee \bigwedge S^n$ is a *DNF* of $\bigwedge(\Gamma \cup \{\sim \phi\})$, then

$$\Gamma \cup \{\sim \phi\} \dashv\vdash \{\textstyle\bigwedge S^1 \vee ... \vee \bigwedge S^n\}$$

and by the decision procedure

$$\not\models \Gamma \cup \{\sim \phi\} \leftrightarrow S^1_\nu = \{\bot\} \wedge ... \wedge S^n_\nu = \{\bot\}.$$

If $\Gamma \cup \{\sim \phi\}$ is unsatisfiable, it follows that $\Gamma \cup \{\sim \phi\} \dashv\vdash \{\bot\}$, since each S^i is deductively equivalent with $\{\bot\}$. From $\Gamma \cup \{\sim \phi\} \vdash \bot$ it follows by the deduction theorem first $\Gamma \vdash \sim \phi \supset \bot$ and thus $\Gamma \vdash \sim \bot \supset \phi$. From $\Gamma \vdash \sim \bot \supset \phi$ and $\Gamma \vdash \sim \bot$ by MP then $\Gamma \vdash \phi$. □

3.16. COROLLARY. *For every finite set of formulas Γ and each formula ϕ, $\Gamma \vdash \phi$ is decidable.*

PROOF. By the completeness and soundness we know $\Gamma \vdash \phi \leftrightarrow \Gamma \models \phi$. Since ϕ is a logical consequence of Γ, iff $\not\models \Gamma \cup \{\sim \phi\}$, we can decide $\Gamma \vdash \phi$ by the procedure for deciding $\not\models \Gamma \cup \{\sim \phi\}$. □

Acknowledgments

The author has been supported during the writing of the submitted draft version of this paper by the EEC Esprit project

DYANA at the Institut für maschinelle Sprachverarbeitung, Universität Stuttgart. The author would like to thank Jochen Dörre, Mark Johnson, Hans Kamp, Ron Kaplan, Paul King, John Maxwell and Stefan Momma as well as an anonymous reviewer for their comments on earlier versions of this paper. All remaining errors are of course my own.

References

[Beierle/Pletat 88] Beierle, C., U. Pletat: Feature Graphs and Abstract Data Types: A Unifying Approach. *Proceedings of COLING 88*, Budapest 1988

[Church 56] Church, A.: Introduction to Mathematical Logic, Princeton 1956

[Dawar/Vijay-Shanker 90] Dawar, A., K. Vijay-Shanker: An Interpretation of Negation in Feature Structure Descriptions. *Computational Linguistics*, 16, 11-21

[Gallier 87] Gallier, J.: Logic for Computer Science: Foundations of Automatic Theorem Proving, New York 1987

[Johnson 88] Johnson, M.: Attribute-Value Logic and the Theory of Grammar. PhD thesis, Stanford University, Stanford 1988

[Kasper/Rounds 86] Kasper, R., W. Rounds: A Logical Semantics for Feature Structures. *Proceedings of the 24th Annual Meeting of the ACL*, New York 1986

[Kasper/Rounds 90] Kasper, R., W. Rounds: The Logic of Unification in Grammar. *Linguistics and Philosophy*, 13, 35-58

[Knuth/Bendix 70] Knuth, D., P. Bendix: Simple Word Problems in Universal Algebra. In: J. Leech (ed.): *Computational Problems in Universal Algebra*, Elmsford, N.Y., 1970

[Kreisel/Tait 61] Kreisel, G., W. Tait: Finite Definability of Number-theoretic Functions and Parametric Completeness of Equational Calculi. *Z. Math. Logik Grundlagen Math.*, 7, 28-38.

[Langholm 89] Langholm, T.: How to Say No with Feature Structures. *COSMOS-Report No. 13*, Oslo 1989

[Nelson/Oppen 80] Nelson, G., D. Oppen: Fast Decision Procedures Based on Congruence Closure. *ACM*, 27, 356-364

[Smolka 89] Smolka, G.: Feature Constraint Logics for Unification Grammars. *IWBS Report 93*, Stuttgart 1989

[Statman 74] Statman, R.: Structural Complexity of Proofs. PhD thesis, Stanford University, Stanford 1974

[Statman 77] Statman, R.: Herbrand's Theorem and Gentzen's Notion of a Direct Proof. In: J. Barwise (ed.): *Handbook of Mathematical Logic*, Amsterdam 1977

[Wedekind 90] Wedekind, J.: Unifikationsgrammatiken und ihre Logik. PhD thesis, Universität Stuttgart, Stuttgart 1990