Logic and Feature Structures*

Mark Johnson
Cognitive and Linguistic Sciences, Box 1978
Brown University
U.S.A.

Abstract

Feature structures play an important role in linguistic knowledge representation in computational linguistics. Given the proliferation of different feature structure formalisms it is useful to have a "common language" to express them in. This paper shows how a variety of feature structures and constraints on them can be expressed in predicate logic (except for the use of *circumscription* for non-monotonic devices), including sorted feature values, subsumption constraints and the non-monotonic *ANY* values and "constraint equations". Many feature systems can be completely axiomatized in the *Schonfinkel-Bernays class* of first-order formulae, so the decidability of the satisfiability and validity problems for these systems follows immediately.

1 Introduction

The number of different feature structure devices and formalisms proposed in the "unification grammar" literature is growing so fast that it is important to find a "common language" in which they can be expressed. This paper shows how a variety of different types of feature structures and constraints on them can be expressed in predicate logic (or standard non-monotonic extensions thereof), including:

- (parameterized) sorts,
- subsumption constraints, and
- non-monotonic constraints (specifically LFG "constraint equations").

These were chosen merely to give a feeling for the ease with which fairly complex constructions can be described in predicate logic; they by no means exhaust the class of feature structures and constraints that can be given first-order axiomatizations. This suggests that instead of requiring radically new types of interpretations, even complex feature structures and constraints on them can be described using standard techniques. Not all these extensions have to be used in an implementation, of course, a reasonable policy might be to only implement the decidable extensions described below, for example. Since feature structures are a kind of frame representation, the results presented here should be of interest to the wider A.I. community.

The results in this paper extend those presented in earlier work, especially [8,9,10,20,21], with which (due to space limitations) familiarity is presupposed.

2 The Schonfinkel-Bernays Class

One advantage of axiomatizing feature structures in first-order predicate logic is that its proof-theoretic, model theoretic and computational properties are well-known. This paper exploits some of the results on *dccidable classes* of first-order logic [3,5] to show that the satis fiability and validity problems for certain types of feature structure constraints are decidable. We show that variety of such structures and constraints can be axiomatized using formulae from the Schonfinkel-Bernays class, (called *SB* below); since the satisfiability and validity of any formula in *SB* is decidable, the satisfiability and validity problems for these feature structure constraints must be decidable too.

A formula is in *SB* iff it is of the form

$$\exists x_1, \ldots, x_m \forall y_1, \ldots, y_n \phi$$

where ϕ is a formula containing no function or quantifier symbols.[1] *SB* formulae possess the *finite model property,* i.e. if a formula has a model then it has a finite model [7]. Lewis [6] investigated the computational complexity of the decision problem for such formulae and showed that the satisfiability and validity problems for *SB* formulae are PSPACE-complete, and (by Proposition 3.2 of [6]) that the satisfiability and validity problems for SB_n are NP-complete, where SB_n is the class of *SB* formulae con taining n or fewer universal quantifiers.

*The research described here was conducted while I was a Gastprofcssor at the Institut fur maschinelle Sprachverarbeitung, Universitat Stuttgart, which I would like to thank for providing a supportive environment in which to get this work done. Robert Garpenter, Jochen Dorre, Andreas Eisele, John Maxwell, Michael Moorreau and Gert Smolka have all made insightful comments on the work reported here; naturally all responsibility for errors remains my own.

[1] Because ϕ can always be expanded to disjunctive normal form, Schonfinkel-Bernays formulae are a generalization of Datalog clauses that allow disjunctive consequents. I hope to explore this more fully in later work.

The conjunction (and disjunction) of two *SB* formulae is itself a *SB* formulae (after quantifiers have been moved outward). This means that *SB* axiomatizations of different kinds of feature structures can be conjoined, and the satisfiability of the composite system is also decidable. For example, Johnson [10] axiomatizes 'set values* using *SB* formulae, and these axioms can be combined with e.g. the *SB* axioms for sorts given below to yield a system with both 'set-valued' and 'sorted' objects completely axiomatized in *SB,* and hence with a decidable satisfiability problem.

3 Attribute-value features

Attribute-value features, popularized in the work of Shieber [18] and others, are the most common type of features used in computational linguistics. The treatment here is effectively the same as that in [10,20,21], so it is only summarized here.

Attribute-value features are in effect *partial functions (SEE.* [12,8]), and they can be formalized in first-order logic either

- as relations that obey functionality axioms (as is done here), or
- as "total" functions over a domain that includes a designated element that is interpreted as an "undefined" value (sec [8,9]).

These two different formalizations reduce the satisfiability problem for attribute-value feature structure constraints to different classes of first-order formulae, and lead to different insights about the original problem.

Here, we conceptualize "pure" attribute-value features as instances of an "arc" relation, where *arc(x,a,y)* is true iff there is an arc labelled *a* from *x* to *y.* (In some expositions the elements *x* and *y* are called *nodes* and *a* is called an *attribute).* The following axioms express the constraints that the *arc* relation must satisfy for it to be an attribute-value structure. The predicate *con* is true of the attribute-value constants.

The first axiom requires that the *are* relation is functional. (Used as an inference rule in a forward chaining system, if requires the "unification" of the "destinations" of a pair of arcs labelled with the same attribute leaving the same node; see [9,10] for details).

$$\forall x, a, y, z \; arc(x, a, y) \wedge arc(x, a, z) \longrightarrow y = z. \qquad (1)$$

The second axiom requires that requires that attribute-value constants have no attributes. (This axiom is responsible for "constant-compound clashes").

$$\forall c, a, y \; \neg(con(c) \wedge arc(c, a, y)) \qquad (2)$$

The attribute-value constants have special properties expressed by the next two axiom schema. These schema stipulate properties of the entities that the attribute-value constants, a subset of the constant symbols of the first-order language, denote. Not every constant symbol will be an attribute-value constant, since it is useful to have constant symbols that refer to other entities as well.

The first schema requires that every attribute-value constant have the property *con.*

For all attribute-value constants c, *con(c).* (3)

The second axiom schema requires that distinct attribute-value constants have distinct denotations. (This is sometimes called a "unique name assumption").

For all pairs of distinct attribute-value constants C_1 and c_2, (4)

$$c_1 \neq c_2.$$

This axiomatization is quite permissive, in that it allows

- cyclic structures,
- infinite structures,
- intensional structures (i.e. different elements may share the same attributes and values),
- disconnected structures, and
- allows values to be used as attributes.

Additional axioms could have been added to prohibit such structures, but because there seems to be no linguistic or computational motivation for such additional stipulations they are not made here. (Axioms prohibiting cyclic structures and attributes from being values can be formulated in *SB,* an *extensionality requirement* can be axiomatized using a first-order formula not in *SB,* while an axiom prohibiting infinite structures cannot be expressed in first-order logic).

Each node in a syntactic parse tree is associated with an element (different nodes can be associated with the same element; see Chapter 3 of Johnson [8] for full details). Lexical entries and syntactic rules constrain the elements associated with parse-tree nodes. Following Kaplan and Bresnan [12] we represent these constraints by *quantifier-free formulae.*

For example, a (simplified) lexical entry for the English verb *swim* might require that:

- the attribute-value element *u* associated with a node dominating the terminal item *swim* have a *semantics* attribute whose value is *swim* (which abbreviates the verb's "semantic value"),
- that *u* have an *agreement* attribute whose value is, say, *v,* and
- that the value of v's *number* and *person* attributes (representing the verb's agreement features) *not* be *singular* and and *3rd* respectively (these are the features of the inflected form *swims).*

These constraints might be expressed in an extended PATR-II notation (see Shieber [18]) as

$$\begin{array}{l} \langle u \, semantics \rangle = swim' \wedge \\ \neg(\langle u \, agreement \, number \rangle = singular \wedge \\ \quad \langle u \, agreement \, person \rangle = 3rd \;) \end{array}$$

and in FDL notation (see Smolka [20,21]) as

$$\begin{array}{l} semantics : swim' \wedge \\ \neg(agreement : number : singular \wedge \\ \quad agreement : person : 3rd \;) \end{array}$$

This paper takes no position on what *notation* such feature constraints should be written in, but simply suggests that whatever notation is used to express this constraint it should *mean* the same thing as the following

(universal) *quantifier-free* formula.

$$\begin{aligned}\exists v \quad & arc(u, semantics, swim') \wedge \\ & arc(u, agreement, v) \wedge \\ & \neg(arc(v, number, singular) \wedge \\ & \quad arc(v, person, 3rd))\end{aligned}$$

(In this formula *u* is a constant that is not an attribute-value constant, while *semantics , swim, number, singular, person* and *3rd* are attribute-value constants). Arbitary boolean combinations (including negation) of attribute-value constraints can be expressed in this manner.[2]

Note that the axioms defining attribute-value features and formulae expressing attribute-value constraints are all in *SB,* so their conjunction is also (equivalent to) a *SB* formula, and hence the satisfiability of such systems of constraints is decidablc. Further, the only quantifiers appear in the axioms 1 and 2 so this conjunction is infact in SB_7, and hence the satisfiability problem for systems of feature constraints is in NP.[3] Since the satisfiability problem for arbitrary conjunctions and disjunctions of atomic feature structure constraints (here, *arc* atoms) is NP-complete [13], the satisfiability problem for the system described here is NP-complete.

4 Sorts

The term "sort" is used to mean different things by different authors. We sketch here how two common interpretations of sorts can be axiomatized by considering some simple examples, and follow Smolka [20,21] in modelling sorts by unary predicates *(parameterized* sorts are modelled by predicates of higher arity).

First,, suppose that sorts are taken to restrict the possible attributes of an element, so that e.g. something of sort *agr* can only have the attributes *number* or *person* (with perhaps restrictions on the values of these attributes that for simplicity are not dealt with here). The following axiom defines this notion of sort.

$$\begin{aligned}&\forall x \; agr(x) \longrightarrow \\ &\quad \forall a, y \; arc(x, a, y) \rightarrow a = person \vee a = number.\end{aligned} \tag{5}$$

(This one place predicate *agr* could be used in a more refined lexical entry for *swim* that requires *agr{v),* i.e. that *v* be of sort *agr.)*

Second, suppose that sorts are also to require that certain attributes *must* have a value, so that e.g. something of sort *agr*[1] must have values for the attributes *number* or *person.* (Axiom 5 only prevents anything satisfying *agr* from having any attributes other than *person* and *number).* The following axiom defines this sort.

$$\begin{aligned}&\forall x \; agr'(x) \longrightarrow \\ &\quad \exists y, z \; arc(x, person, y) \wedge arc(x, number, z).\end{aligned} \tag{6}$$

[2]The proper treatment of negation in "feature logics" has been the subject of considerable discussion [15,16,1,8,9,10]: however I know of no linguistic application in which a classical interpretation of negation yields intuitively "incorrect" results.

[3]The axioms 1 and 2 can be replaced with an equivalent axiom that "shares" the universally quantified variables, so systems of attribute-value constraints can be expressed as formulae in SB_4.

Both kinds of sorts can be optionally augmented by an *extensionality requirement,* which stipulates that no two distinct elements in the same sort have identical values for their attributes. For example, the following axiom requires that no two elements of sort *agr* can have the same values for their *person* and *number* attributes.

$$\begin{aligned}\forall x, y, u, v \quad & (agr(x) \wedge arc(x, number, u) \wedge \\ & \quad arc(x, person, v) \wedge agr(y) \wedge \\ & \quad arc(y, number, u) \wedge \\ & \quad arc(y, person, v)) \\ & \quad \rightarrow x = y\end{aligned} \tag{7}$$

Because axioms of the form of 5 and 7 are in *SB,* the satisfiability of attribute-value systems augmented with sort constraints of the first type is decidable, and (given a fixed set of sorts and hence a fixed number of universal quantifiers) is NP-complete. On the other hand, axioms of the form of 6 are not in *SB.* While this does not imply undecidability, Smolka [20,21] has shown that for systems that allow *parameterized sorts* (i.e. sort predicates with arity greater than one) this is infact the case.

Of course, there is no reason to restrict attention to unary predicates. For example, assuming that lists are represented in the standard attribute-value formulation using *first* and *rest* attributes (see Shieber [18] or Johnson [8] for details), the following axioms define the predicate *membcr(x,l),* which is true iff *x* appears somewhere in the proper list represented by *l.*[4]

$$\begin{aligned}\forall x, l \; arc(l, first, x) &\longrightarrow member(x, l) \\ \forall x, l, l' \; arc(l, rest, l') \wedge & \\ member(x, l') &\longrightarrow member(x, l) \\ \forall x, l, l' \; member(x, l) &\longrightarrow arc(l, first, x) \vee \\ & \quad (arc(l, rest, l') \rightarrow \\ & \quad\quad member(x, l'))\end{aligned} \tag{«}$$

Again, since the axioms in 8 are in *SB,* they can be con joined with any other *SB* axioms and the system remains decidable.

5 Subsumption Constraints

This section and the next focusses on some of the most difficult aspects of the theory of feature structures. Subsumption constraints are notoriously tricky to formulate. Partly, this is because the term 'subsumptiori' is used to refer to two different notions in the feature structure literature.

First, subsumption can be construed as a relation between system of constraints. ϕ *subsumes U/* iff every feature structure that satisfies ψ also satisfies ϕ. (This notion of subsumption is used in the prediction step of generalized Earley and LR parsing algorithms for feature structure grammars, see Shieber [19,17] for details.) In the framework described here, ϕ and ψ are both

[4]The axioms in 8 do not require that / be (an attribute-value encoding of) a list. A unary 'sort' predicate that docs require this is easily formulated, however. Among other things, this predicate should require that the "empty list" constant *nil* has neither *first* nor *last* arcs leaving it. (This could also be achieved by treating *nil* as an attribute-value constant).

quantifier-free formulae (e.g. boolean combinations of *arc* atoms), so ϕ subsumes ψ iff

$$A \models \psi \rightarrow \phi$$

where *A* is the relevant feature-structure axiomatization. Clearly, if *A* is in *SB* then this notion of subsumption is decidable.

Second, subsumption can be construed as a relation between elements within a feature structure, where e subsumes e', written $c \sqsubseteq c'$ iff there is a partial en-(lornorphism *h* such that *h(c)* = e', that preserves attribute value constants and attributes and their values (and possibly sorts). (This notion of subsumption is needed to describe the agreement agreement properties of conjoined phrases; see Shieber [19] for details.) It is straight-forward to axiomatize this in second-order predicate logic by treating the partial endomorphism *h* as a functional relation (i.e. /i(x, *y)* iff *h(x)* is defined and equal to *y).*

$$\begin{aligned}&\forall c, c'\ c \sqsubseteq c' \longleftrightarrow \\ &\quad \exists h\ (\ \forall x, y, z\ h(x,y) \wedge h(x,z) \rightarrow y = z \wedge \\ &\qquad h(c, c') \wedge \\ &\qquad \forall c\ con(c) \rightarrow h(c,c) \wedge \\ &\qquad \forall x, y, a, u\ h(x,y) \wedge arc(x,a,u) \rightarrow \exists v\ arc(y,a,v)\ )\end{aligned}$$

Dorre and Hounds [2] have shown the undecidability of conjunctions of subsumption and attribute value constraints, so clearly this notion of subsumption cannot be axiomatized in *SB.* Perhaps surprisingly, *positively occuring* subsumption constraints[5] can be axiomatized quite directly in first-order logic in a manner discovered jointly with John Maxwell.

As just formulated, subsumption seems to rely on an existential quantification over partial endomorphisms h, but by

- replacing the biconditional with an implication (which does not affect satisfiability if all subsumption constraints occur positively), and
- skolemizing the embedded existential quantification

we obtain an equivalent formulation in terms of a four-place relation *h',* where (e, e', x, *y)* iff h(x, y), where *h* is the partial endomorphism whose existence is asserted by the existential in the definition of $c \sqsubseteq c'$. The first axiom has the same effect as requiring that $h(c) = c'$.

$$\forall c, c'\ \ c \sqsubseteq c' \longrightarrow h'(c, c', c, c').$$

The second axiom requires that *h'* preserve attributes and their values.

$$\begin{aligned}&\forall c, c', x, y, a, z\ h'(c, c', x, y) \wedge arc(x, a, z) \\ &\quad \longrightarrow \exists v\ arc(y, a, v) \wedge h'(c, c', z, v).\end{aligned} \tag{10}$$

The third axiom requires that h' preserve constants.

$$\forall c, c', y\ h'(c, c', c, y) \longrightarrow c = y. \tag{ii}$$

The fourth axiom requires that *ti* is functional.

$$\begin{aligned}&\forall c, c', x, y, z\ h'(c, c', x, y) \wedge h'(c, c', x, z) \\ &\quad \longrightarrow y = z.\end{aligned} \tag{12}$$

[5]A subformula occurs positively iff it is in the scope of an even number of negation symbols. The simplification of a biconditional to an implication when the relation defined by the biconditional appears elsewhere only positively is described and proved not to alter satisfiability in [10).

6 Constraint equations and *ANY* values

Finally, we turn to perhaps the most thorny problem for any theoretical account of feature structures: default values and other non-monotonic constructions. This section shows how these notions can be formalized by using *circumscription* to require satisfying models to be *minimal models*[6]. This approach has two major advantages over other approaches:

- expansion to disjunctive normal form is not required, and
- a single notion of satisfiability is defined which treats the monotonic and nonmonotonic construe tions simultaneously.

Several versions of circumscription are discussed in the literature; for an introduction see e.g. Genesereth and Nilsson [4]. The *parallel circumscription formula <f>'* for relations $R_1, \ldots, R_n$ in ϕ has the property that a model $\mathcal{M}$ satisfies $\phi \wedge \phi'$ iff $\mathcal{M}$ is an $R_1, \ldots, R_n$-minimal model of ϕ. (In general *</>'* is a second-order formula).

An important intuition guiding early work in unification grammar (especially that of Kaplan and Bresnan [12] and Kay [14]) is that only the *minimal* feature structures satisfying the constraints are of linguistic in terest, and that lexical entries and syntactic rules may impose additional conditions that a minimal model has to satisfy in order to be well-formed. This section shows how these intuitions can be formalized using circumscription.

For example, most current theories of natural language syntax posit a requirement that all noun-phrases must be *assigned* a 'case feature' by some syntactic rule or lexical entry. This could be implemented in a feature-structure based system by adding a constraint to all lexical entries for nouns that a *minimal* model is well-formed only if the associated feature element has a case attribute; this is sometimes called an *ANY*-value constraint on the *case* attribute. Similarly, a *constraint equation* between two entities $x =_c y$ is satisfied iff *x* = *y* in a minimal model of the attribute-value formulae. (See the discussion on pages 108-110 of Johnson [8] for a more detailed explanation of such constraints.)

"Constraint equations" and *ANY* values can be treated in the following way. We represent the constraint that an attribute *a* must be defined on an element *x* in a minimal model by *any(x,a),* and constraint equations by $x =_c y$. Now let ϕ be the conjunction of the equality axioms, the attribute-value axioms and all of the (formulae corresponding to) feature structure con straints from a given parse, and let ϕ' be the parallel circumscription formula for arc, *con* and — in ϕ. We circumscribe precisely these relations because a minimal model is one which possesses as few ares as possible, specifies attribute-value constants as the denotation of as few variables as possible, and identifies as equal or "unified" as few pairs of variables as possible (see the definition of the subsumption ordering on attribute-value models in [8]).

[6]Fernando Pereira suggested to me that circumscription could be used to provide a formal account of non-monotonic feature structure constraints.

Then a model *M.* satisfies all of the constraints (including the so-called "defining equations", the "constraint equations" and the *ANY* constraints) iff

$$\begin{aligned} \mathcal{M} \models\ & \phi \wedge \phi' \wedge \\ & \forall x, a\ any(x, a) \leftrightarrow \exists y\ arc(x, a, y) \wedge \\ & \forall x, y\ x =_c y \leftrightarrow x = y. \end{aligned} \tag{13}$$

The circumscription of equality requires that two constants denote the same entity (i.e. are "unified") in a model ifr interpreting them as denoting distinct entities would result in the violation of some axiom or constraint. The circumscription of *arc* and *con* requires that these relations are also minimal.

Note that this formulation restricts attention to "clas sical" minimal models. However, for some applications this seems to be too strong. For example, the constraint attached to the NP child in the LFG rule [12]

$$\begin{array}{lcc} \text{VP} \longrightarrow & \begin{array}{c}\text{V}\\ \uparrow{=}\downarrow\end{array} & \begin{array}{c}\text{NP}\\ \uparrow \text{OBJ} = \downarrow\\ (\downarrow \text{CASE} = \text{ACC})\end{array} \end{array}$$

includes an *optional* feature structure constraint, which would be represented in the framework described here as

$$arc(vp, \text{OBJ}, np)\ \wedge\ (arc(np, \text{CASE}, \text{ACC}) \vee true)$$

Now, the left-hand disjunct contributes nothing to the truth conditions if disjunction is interpreted classically (since $\phi \vee true = true$), so this is clearly not the intended interpretation. Rather, Kaplan and Bresnan seem to interpret disjunction as a kind of non-deterministic choice operator, so that all of the minimal models of both ϕ and ψ are also minimal models of $\phi \vee \psi$.

7 Conclusion

This paper has shown how a wide variety of different types of feature structures and constraints on them can be described using predicate logic. The decidability of the satisfiability problem of many interesting feature structure systems follows directly from the fact that they can be axiomatized in the Schonfinkel-Bernays class. Further, axiomatizing feature structures in first-order logic allows us to apply standard techniques to the formalization of nonmonotonic feature structure constraints.

References

[1] Dawar, A. and K. Vijay-Shanker. 1990. "An Interpretation of Negation in Feature Structures" in *Computational Linguistics* 16.1, pages 11 21.

[2] Dorre, J. and W. Rounds. 1989 *On Subsumption and Scmiunification in Feature Algebras.* IWBS Report 97. IBM Deutschland.

(3] Dreben and Goldfarb. 1979. *The Decision Problem: Solvable Classes of Quantificational Formulas.* Addison-Wesley.

[4] Genesereth, M. and N. Nilsson. 1987. *Logical Foundations of Artificial Intelligence.* Morgan Kaufmann, Los Altos, California.

[5] Guerevyich, Y. 1976. "The Decision Problem for Standard Classes", in *The Journal of Symbolic Logic,* 41.2, pages 460-464.

[6] Lewis, H. 1980. "Complexity Results for Classes of Quantificational Formulae", in *Journal of Computer and System Sciences,* 21, pages 317 353.

[7] Lewis, H. and C. Papadimitriou. 1981. *Elements of the Theory of Computation.* Prentice-Hall.

[8] Johnson, M., 1988. *Attribute-Value Logic and the Theory of Grammar.* CSLI Monographs, Chicago University Press.

[9] Johnson, ML, 1990. "Expressing Disjunctive and Negative Feature Constraints in Classical First-order Logic", in *Proceedings of the 24th Annual Meeting of the ACL.* Pittsburgh.

[10] Johnson, M., "Features and Formulae", to appear in *Computational Linguistics.*

[11] Karttunen, L. 1984. "Features and Values", in *Proceedings of COLING-84-* Stanford University.

[12] Kaplan, R. and J. Bresnan. 1982. "Lexical-functional grammar, a formal system for grammatical representation", in Bresnan, ed., *The Mental Representation of Grammatical Relations.* The MIT Press.

[13] Kasper, R. and W. Rounds. 1990. "The Logic of Unification in Grammar" in *Linguistics and Philosophy,* 13.1, pages 35 58.

[14] Kay, M. 1985 "Unification in Grammar" in V. Dahl and P. SaintDizier, eds., *Natural Language Understanding and Logic Programming.* North Holland.

[15] Moshier, D. and W. Rounds. 1987. "A logic for partially specified data structures" in *The ACM Symposium on the Principles of Programming Lan guages.* Association for Computing Machinery, Munich, Germany.

[16] Pereira, F. "Grammars and Logics of Partial Information" in *The Proceedings of the International Conference on Logic Programming,* The MIT Press.

[17] Shieber, S., 1985. *Using Restriction to extend Pars ing Algorithms for Complex-feature based gram mars.*

[18] Shieber, S., 1986. *An Introduction to Unification-Based Theories of Grammar.* CSLI Monographs, Chicago University Press.

[19] Shieber, S., 1989. *Parsing and Type Inference for Natural and Computer Languages.* SRI Technical Note 460. Palo Alto, California/,

[20] Smolka, G., 1988. *A Feature Logic with Subsorts.* LILOG Report 33, IWBS, IBM Deutschland.

[21] Smolka, G., 1989. *Feature Constraint Logics for Unification Grammars.* IWBS Report 93," IWBS, IBM Deutschland.