

Logic and attribute-value grammar

Anders Søgaard
Center for Language Technology
Njalsgade 80
DK-2300 Copenhagen
anders@cst.dk

May 31, 2006

0.1 Introduction

Central question:

Can we formally characterize the set of possible human languages? In terms of attribute-value grammars?

Our methodology:

The NLS are the efficiently communicative and learnable ones.

Questions that pop up:

- What does it take to be communicative?
- How efficient is efficient?
- Is there such a thing as UG?

The quite ambitious goal of my thesis is to define a (model-theoretic) tractable and learnable attribute-value formalism of reasonable expressive power. Today's talk is primarily about efficiency. The result that I present is an NP-complete fragment of, I argue, adequate power. I show how tractability can also be obtained.

0.2 Historical background

- (1) He is **both** a painter **and** a linguist.
- (2) A white male (whom a white male)ⁿ hiredⁿ hired another white male.
- (3) mer em Hans.DAT es huus.ACC hälfed.DAT aastrichte.ACC.
- (4) govel-i.NOM igi.NOM sisxl-i.NOM saxl-isa-j.GEN.NOM m-is.GEN
Sail-is-isa-j.GEN.GEN.NOM ('all (the) blood.NOM (of the) house.GEN
(of) Saul.GEN)
- (5) dat Jan Piet Marie Fred^k (horde leren^k uitnodigen)⁺ en zag leren^k omhelzen.

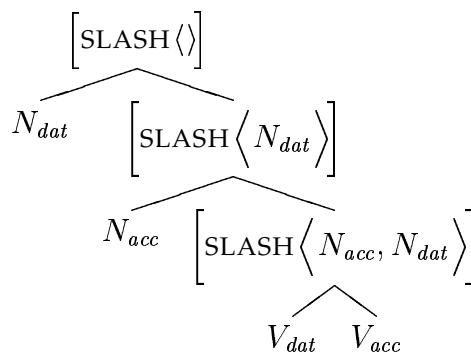
Example 0.1. Let A be the start symbol of

- $A \rightarrow a, A$
- $A \rightarrow a, b$

which then generates $\{a^+b\}$ and corresponds to

- $\left[\begin{array}{l} \text{TYPE } non\text{-terminal} \\ \text{CAT } A \end{array} \right] \rightarrow \left[\begin{array}{l} \text{TYPE } terminal \\ \text{PHON } a \end{array} \right], \left[\begin{array}{l} \text{TYPE } non\text{-terminal} \\ \text{CAT } A \end{array} \right]$
- $\left[\begin{array}{l} \text{TYPE } non\text{-terminal} \\ \text{CAT } A \end{array} \right] \rightarrow \left[\begin{array}{l} \text{TYPE } terminal \\ \text{PHON } a \end{array} \right], \left[\begin{array}{l} \text{TYPE } terminal \\ \text{PHON } b \end{array} \right]$

Unification-based CFGs extend CFGs, however. Reentrancies allow for generalizations (e.g. agreement), and information can be stored (as in indexed languages):



0.3 Parsing as satisfiability

$$\begin{aligned}\mathcal{L}(\mathcal{G}) &= \{\sigma \in \mathcal{V}^* \mid \exists M \in \mathcal{M} \text{ start}' \sqsubseteq M \wedge M \xrightarrow{*} \sigma\} \\ \mathcal{L}(\mathcal{G}) &= \{\sigma \in \mathcal{V}^* \mid \exists M \in \mathcal{M} M, w \models (\text{Axioms} \wedge \sigma')\}\end{aligned}$$

The advantages of model-theoretic algorithms include:

- declarativity,
- scalability,
- (easy) integration,
- encodes weak precedence in a very obvious way,
- flexible grammaticality,
- an open lexicon, and
- an agnostic stand on the cardinality of NLS.

The advantages of logical specification (reconstruction) include:

- formalization,
- comparison (via interlingua), and
- property inheritance.

Example 0.2. A trivial example of property inheritance: The time complexity of some logic is the upper bound on the time complexity of any theory specified in that logic.

Example 0.3. Equally trivial is it that any theory encoded in first order logic is compact.

Example 0.4. The context-free grammar in the above can be rewritten as logical axioms:

$$\begin{aligned}
 \left[\text{TYPE } \textit{non-terminal} \right] &\Rightarrow \left[\begin{array}{l} \text{CAT A } \boxed{1} \\ \text{LEFT } \left[\begin{array}{l} \text{TYPE } \textit{terminal} \\ \text{PHON a} \end{array} \right] \\ \text{RIGHT } \left[\begin{array}{l} \text{TYPE } \textit{non-terminal} \\ \text{CAT } \boxed{1} \end{array} \right] \end{array} \right] \vee \\
 &\left[\begin{array}{l} \text{CAT A} \\ \text{LEFT } \left[\begin{array}{l} \text{TYPE } \textit{terminal} \\ \text{PHON a} \end{array} \right] \\ \text{RIGHT } \left[\begin{array}{l} \text{TYPE } \textit{terminal} \\ \text{PHON b} \end{array} \right] \end{array} \right]
 \end{aligned}$$

0.4 Complexity of universal recognition

HPSG	Undecidable	[Car91]
LFG	Undecidable	[Joh88]
CUG	NP-complete	[Tra95]
CFG	$\mathcal{O}(n^3)$	[Ear70]

CUG has some obvious disadvantages. It does not encode free word order, scrambling and discontinuous constituency in any obvious way; it allows no unary extensions, and it cannot quantify over attribute paths, i.e. it does not have anything like LFG-style functional uncertainty.

CFG does not capture cross-serial dependencies and other linguistic phenomena. The number of rules tend to explode in applications; for instance, CFGs do not let the linguist generalize over agreement features.

[Joh88] defines an NP-complete fragment of LFG that does not allow detours, i.e. cyclic unary projections, and which does not incorporate functional uncertainty. This fragment is almost equivalent to the NP-complete HPSG fragment in [Tra95] (in fact, it is a bit weaker, since it does not allow for inheritance).

0.5 Categorical Unification Grammar

We present a new proof of NP-completeness for CUG, different from the one obtained in [Tra95]. CUG is to A/B grammars what PATR is to CFG. You may recall that an A/B grammar consists of a set of two combinatory rules:

$$\begin{aligned} X(X/Y) &\rightarrow Y \quad (\text{left application}) \\ (Y/X)X &\rightarrow Y \quad (\text{right application}) \end{aligned}$$

The simple move into CUG is to replace the syntactic categories with attribute-value structures and turn each production step into a two-step procedure of *instantiation* and *stripping*, where instantiation is just unification, and stripping amounts to a kind of resource-sensitive follow-up, i.e. it removes the saturated Xes (in the above) from the syntactic category of the mother of the functor-argument structure.

Since only the constituent structure is recursive, and since all rules are binary, the smallest model (the shortest derivation) of a string σ is at most $(2^{|\sigma|}-1) \times \text{paths}$. This is in fact enough to establish an NPTIME result. The general lemma that I'll employ in this talk can be formulated as [BdRV01]:

Lemma 0.5. *If Λ (a consistent, normal modal logic) has the polysize model property (pmp), i.e. the smallest model of a formula is at most polynomial in its size, and if model-checking can be solved in PTIME, then satisfiability of Λ is in NPTIME.*

In fact, any consistent, normal modal logic that has the pmp and PTIME model-checking is NP-complete, since any such logic by definition encodes propositional satisfiability. If a theory, say of natural language syntax, can be formulated in such a language, i.e. the problem can be reduced to modal satisfiability, it is in NPTIME, but not necessarily NP-complete.

CUG can be encoded in a fragment of Kasper-Rounds logic with implication and negation (and even simpler languages), where attributes are unary modalities, values are propositions, and derivations are Kripke structures. Since this extended Kasper-Rounds logic is a consistent, normal modal logic with a PTIME model-checking problem [Lan06], and since CUG ensures the polysize model property (see above), it follows that

Theorem 0.6. *The universal recognition problem of CUG can be solved on a polynomial time bound non-deterministic Turing machine.*

0.6 Generalization of our proof

Our proof relies only on the size of the smallest model that satisfies a string. It can easily be derived from this that similar results can be obtained for linearization-based and unordered grammars. Why? Consider first standard rules as they are represented in modal logic. The $\langle\langle\rangle$ -modality encodes immediate precedence.

$$xy_phrase \rightarrow \langle\text{down}\rangle(x \wedge \langle\text{right}\rangle y \wedge \langle\langle\rangle y)$$

This rule enforces the x - and y -constituents to immediately precede each other. If you remove the $\langle\langle\rangle$ -constraint, this is no longer ensured, and the constituents can freely intermingle with others. However, no additional structure can be introduced, if the derivation (Kripke structure) is connected. Consequently, the proof of solvability in NPTIME still applies.

The NPTIME proof does not apply when unary extensions are allowed, since cyclic unary extensions can make the derivations arbitrarily large. If unary extensions are restricted to be *acyclic*, however, the smallest model of a satisfiable string is at most:

$$(2|\sigma| - 1) \times (u + 1) \times \mathbf{paths}$$

where u is the number of unary rules. Consequently, the NPTIME proof still applies.

0.7 Typed feature structure grammar

Definition 0.7. A TFSG is defined on a signature of labels and atoms. Attribute-value structures are acyclic, connected and rooted, and deterministic. Non-wellfounded sets are allowed. The only recursive part of the attribute-value geometry is what declares the functor-argument structure (as in CUG). The language of a TFSG consists of all the strings whose logical descriptions are satisfiable in conjunction with the grammar (axioms).

Some additional properties:

- TFSG encodes linearization (in line with HPSG proposals),
- TFSG has a rigid lexicon (underspecification), and
- TFSG is off-line parsable (unary extensions are acyclic). [This condition can be directly encoded in dynamic logic, but then compactness is lost.]

Corollary 0.8. *TFSG is in NPTIME.*

First order definability

Theorem 0.9. *TFSG, as it is defined here, cannot be embedded in any decidable prefix-vocabulary class of first order logic.*

Proof. The minimal pvc that defines TFSG is $[\forall^3\exists^2]_=$. □

Theorem 0.10. *TFSG can be defined in first order logic, if path quantification is simulated via transitive relations, and the acyclicity constraint is external.*

Corollary 0.11. *TFSG is compact on this set-up.*

0.8 Tractability

Theorem 0.12. *TFSG is NP-complete.*

Proof. Corollary 0.8 says that TFSG is in NPTIME. It is also NP-hard, since TFSG encodes 3SAT. Simply let propositional variables be words, unary rules assign truth values, such that, for instance

$$\begin{array}{c} \left[\begin{array}{l} \text{PHON} \langle p \rangle \\ \text{CONT} \left[\begin{array}{l} P \quad \top \end{array} \right] \end{array} \right] \\ | \\ \left[\begin{array}{l} \text{PHON} \langle p \rangle \\ \text{CONT} \left[P \right] \end{array} \right] \end{array}$$

In other words, each propositional variable has a specific feature in which its truth value is assigned. Clauses are build by ternary phrases that coindex its CONT value with that of one of its daughters. When larger clauses are build the CONT values of the daughter clauses are coindexed with the CONT value of the mother. This ensures consistency. \square

Theorem 0.13. *The rigid and k -ambiguous (non-trivial) fragment of TFSG with regular deepstructures is in PTIME.*

Proof. On a generative perspective, a ID/LP format makes it necessary to check an exponential number of projections (unlike CFGs). If signs combine unambiguously, this number is reduced to

$$\frac{c^2-c}{2} + \sum_{1 < i < c} (k \times g(\frac{c}{2}) \times (c - i)) \text{ where } g(x) \text{ is the biggest natural number smaller than or equal to } x.$$

By unification (whose time complexity is polynomial because the involved feature geometry is non-recursive), the (linear) set of candidate models is restricted to $k \times g(\frac{c}{2})$. This set of models can be evaluated in polynomial time. \square



0.9 Learnability

In the late sixties, already, Gold [Gol67] proved that not even the regular languages are identifiable in the limit from positive data. Several things can (reasonably) be hypothesized on this background: (i) The Innateness Hypothesis is true, (ii) L1 acquisition is guided by negative data (in a way that psychologists have not yet noticed), or (iii) natural languages cross-cut the Chomsky Hierarchy. (ii) is widely conjectured to be false. Universal grammar (UG) can be implemented in many ways, e.g. as operations on a type hierarchy [Vil02] or as feature cooccurrence restrictions.

UG-	
<hr/>	
<i>k</i> -reversible FSA	[Ang82]
rigid A/B	[Kan98]

UG+	
<hr/>	
CUG	[Vil02]

References

- [Ang82] D. Angluin. Inference of reversible languages. *Journal of the Association for Computing Machinery*, 29:741–765, 1982.
- [BdRV01] Patrick Blackburn, Maarten de Rijke, and Yde Venema. *Modal logic*. Cambridge University Press, Cambridge, England, 2001.
- [Car91] Bob Carpenter. The generative power of categorial grammars and head-driven phrase structure grammars with lexical rules. *Computational Linguistics*, 17(3):301–313, 1991.
- [Ear70] Jay Earley. An efficient context-free parsing algorithm. *Communications of the Association for Computing Machinery*, 26(6):400–8, 1970.
- [Gol67] Mark Gold. Language identification in the limit. *Information and Control*, 10:447–474, 1967.
- [Joh88] Mark Johnson. *Attribute-value logic and the theory of grammar*. CSLI Publications, Stanford, California, 1988.
- [Kan98] Makoto Kanazawa. *Learnable classes of categorial grammars*. CSLI Publications, Stanford, California, 1998.
- [Lan06] Martin Lange. Model checking propositional dynamic logic with all extras. *Journal of Applied Logic*, 4:39–49, 2006.
- [Tra95] Marten Trautwein. *Computational pitfalls in tractable grammar formalisms*. PhD thesis, University of Amsterdam, Amsterdam, the Netherlands, 1995. ILLC Dissertation Series DS-1995-15.
- [Vil02] Aline Villavicencio. *The acquisition of a unification-based generalised categorial grammar*. PhD thesis, University of Cambridge, Cambridge, England, 2002. Technical report UCAM-CL-TR-533.