Implementing a Syntax-Morphology Interface for Athabaskan

Emily M. Bender & Jeff Good

LSA, Oakland CA, 1/7/05

Introduction: Montage (2)

[SLIDE]

This talk is situated as part of a larger project called Montage. The goal of Montage is to produce a suite of software tools to assist linguists in the documentation of underdescribed languages. We aim in particular to support the development of grammars—both documentary grammars and machine-readable ones—while integrating our tools with those produced by other intiatives aiming to support the production of transcribed texts and lexicons.

[SLIDE]

Our overarching goal is to enable ordinary working linguists to make use of computational tools, including particularly the techniques of grammar engineering, without having to become computational linguists or grammar engineers themselves. Among other things, Montage will support the authoring of electronic desriptive grammars, semi-automated grammatical annotation of texts, and wizard-based rapid precision grammar prototyping based on the LinGO Grammar Matrix. For more on Montage and related efforts, please stop by the OLAC information table in ...

In this talk, we focus on the Montage model of morphological analysis and the design of a morphology-syntax interface. We are developing tools intended to work across a vast typological landscape and furthermore intended to be useful when the linguist has only a partial idea of what's going on in a language, so it seems prudent to look before we leap and survey both the range of phenomena found in the worlds' languages and the practice of documentary linguists before building anything. Accordingly, this is a programmatic talk, which we present in the hopes of getting feedback on the proposed design.

Overview

[SLIDE]

Alright, that was the introduction and goals. In the approximately 30 seconds we have left, we'll start with some terminological distinctions, then reject two possible interfaces, before presenting the proposed system. Along the way, we'll illustrate our argumentation with examples from the Athabaskan language Slave. We have chosen Slave because, as an Athabaskan language, it presents a rich array of morphological dilemmas and because it is beautifully documented in the work of Keren Rice. Our intention with keeping a hard case like Slave in mind is to remind ourselves that we want to support work on typologically diverse languages, that we want to support the exploration of alternative hypotheses, and that we want to support the encoding of whatever discoveries the linguist has made, no matter how preliminary or incomplete.

Terminological distinctions

[SLIDE]

Before going any further, we need to set out some terms for a distinction that is going to become important. At the risk of spending half of this talk saying "morpho", we're using the terms "morphophonology" and "morphosyntax" to draw the distinction. *Morphophonology* includes morphotactics, or the possible orderings of morphemes within words, morphologically conditioned phonological alternations, general phonological processes, and furthermore the mapping between surface forms and abstract morphemes.

For roots, we'll take abstract morphemes to be underlying forms. For affixes, we'll take the abstract morphemes to be feature-like pieces of information, like 'plus nominative' or 'plus causative'. In this, we follow the model presented by Beesley and Karttunen 2003.

Morphosyntax, on the other hand, concerns the way in which full syntactic and semantic representations are built up out of analyses of strings of abstract morphemes. Depending on your theory of grammar, this might involve something like lexical rules or subword trees.

Possible interfaces

[SLIDE]

Before describing the possible interfaces, we should note that we are working with HPSG as our syntactic theory and the LKB as our grammar development environment. Furthermore, we are using XFST as our current morphological anlaysis engine. However, we believe that the interface concerns we bring up are largely independent of the particular systems chosen.

We have turned up three possible general interface designs. The first two turn out to be bad ideas. These are trying to do the morphophonology in the morphosyntax and trying to do the morphosyntax in the morphophonology. The third possibility is allowing morphophonology and morphosyntax to be independent.

Morphology in syntax

[SLIDE]

The HPSG incarnation of Morphophonology in Morphosyntax is to have lexical rules which each perform some morphosyntactic operation and at the same time, call on a particular morphophonological function to produce the phonological form for the mother/output. For the most part, HPSG work doesn't consider the nature of those morphophonological functions, with the notable exception of Orhan Orgun's work. Likewise, the current version of the LKB grammar development environment assumes that orthography- or phonology- changing effects are specified as a part of the morphosyntactic rules which add the affixes involved.

[SLIDE]

We believe that this is inefficient, for at least three reasons. First, in descriptive linguistics morphophonological analysis is a prerequisite to morphosyntactic analysis. Thus a documentary linguist working on a language is likely to work out a set of general and morpheme-specific phonological rules before ever starting to think about how the addition of a particular affix affects the syntactic combinatory potential of a root. If that linguist or another linguist ever does move on to developing an implemented syntactic grammar for the language, it would be an additional, unnecessary bit of work to have to have to break apart whatever phonological system is already built, in order to associate pieces of it with various morphosyntactic rules.

As for the second inefficiency, we suspect that farming the phonological functions out across the morphosyntax would preclude effectively merging them into a single one-step analyzer like the efficient ones which can be built using composed finite state transducers.

Finally, this strategy is particularly awkward when it comes to purely phonological effects.

[SLIDE]

An example from Slave is the epenthetic *he-* prefixed to verb stems which would otherwise be initial in their words. To handle this on a morphophonology-in-morphosyntax approach, one would need a lexical rule with no morphosyntactic effects which merely checked the phonological form of all verbs and added the epenthetic prefix in the few cases where it was necessary.

Morphosyntax in morphophonology

[SLIDE]

So much for morphosyntax in morphophonology. The opposite mistake is to look at the underlying forms that a morphological analyzer like a transducer can produce – things like 'eat plus verb plus 3rd person singular plus causative plus past' – and equate those 'plus foo' features with the kind of feature-value pairs that are used in many syntactic systems. The idea would be to generalize one step, and to output 'case nominative' instead of 'plus nominative', perhaps with appropriate brackets around it, and hope that the output could be directly translated to a lexical edge used by the syntactic parser.

The problem here is that this idea doesn't generalize well from the morphosyntactically simple cases like, well, case to the morphosyntactically complicated cases like causatives.

[SLIDE]

The feature structures that are needed for syntax and semantics are much more complicated than bundles of atomic-valued features like 'case nominative'. They have internal structure. They are potentially recursive. And they might involve reentrancy. Morphological causatives are a case in point – the complex set of relationships between semantic relations and between syntactic and semantic arguments that need to be represented are beyond what it is possible to represent, at least conveniently, in a string-based formalism. Why not make the output of the morphological analysis something richer than a string? Our claim is that the sort of formalism required to represent syntactically and semantically useful information is not otherwise required for morphophonology. It would be overkill, and it would almost certainly unnecesarily complicate the formalism a linguist working on morphophonology would have to deal with.

Theoretical conclusion

[SLIDE]

We conclude that morphophonology and morphosyntax are independent yet articulated systems. The point where they meet is abstract morphemes. We have argued for this on the basis of practical considerations, but we feel that it is a reasonable approach on a theoretical level as well. This is in keeping with our general tendency towards bottom-up exploration of grammar: we're happy to assume that they're separate until the evidence shows us we're missing generalizations by doing do.

Independent morphology and syntax

[SLIDE]

In the independent-systems version, the morphophonology's job is to map surface strings to strings of abstract morphemes: things like eat-plus-causative-plus-imperfective-plus-2sgsubject. And vice versa. The job of the morphosyntax is to map those word-sized bundles of abstract morphemes into syntactico-semantic structures which can then be combined by the rules of syntax into phrases, with all the right syntactic and semantic dependencies.

Run-time interface

[SLIDE]

These considerations lead us to the same kind of run-time interface which has been used before in any project where the morphophonological analyzer (usually just called a morphological analyzer) is produced separately from the grammar. In the parsing direction, this typically involves first a preprocessing step which splits the incoming string into words–although it should be noted that tokenization and morphological analysis can be more intermingled in languages like Japanese which don't represent word boundaries orthographically. The tokenization step handles sandhi rules—that is, phonological processes across word boundaries. This is followed by morphophonological analysis which takes the string of words and returns a string of strings of abstract morphemes. The strings of abstract morphemes are then passed to the sublexical component of the parser, which in turn builds syntactic words to be combined into phrases by the parser proper.

Where this proposal differs from existing systems is in suggesting a development interface between morphophonology and morphosyntax in addition to the run-time interface. Unlike in industrial grammar engineering contexts, we believe that grammar engineering for language documentation will typically involve the same individual or small group of individuals working on both the morphophonological and morphosyntactic analysis. To avoid systematic duplication of entries, we propose a single, bipartite lexical database, representing both the morphophonological and morphosyntactic properties of stems.

Bipartite lexical database design

[SLIDE]

As depicted schematically on this slide, the lexical database contains entries for stems which relate them to morphophonological classes (such as Romance conjugation classes or the classes of verbs defined by Slave classifiers which we'll see in a moment), positions in templates, co-phonologies, etc. The intent is that this information could be compiled into a file defining underlying forms for the morphophonology. These forms would be simple concatenations of abstract morphemes and it would be up to the morphophonological analyzer to handle non-concatenative morphology appropriately. At the same time, stems are mapped to the information required by the morphosyntatic analyzer, such as their lexical semantics, their syntactic part of speech, their syntactic valence, etc.

We will build on the lexical database already implemented for the morphosyntactic side in the LKB as well as the FIELD system developed by the EMELD group to assist field linguists in producing lexicons.

Deveolopment interface

[SLIDE]

Working with the lexical database as a development-time interface between morphophonology and morphosyntax, the linguist would be able to enter each abstract stem exactly once. Furthermore, the lexical database will support the definition of 'macros' representing default or common pairings of morphophonological and morphosyntactic classes. Of course, use of the macros wouldn't be required and idiosyncratic pairings would also be expected. Finally, the design of the system will allow one morphosyntactic form to map to multiple morphophonological forms with the same stem and vice versa.

Slave Verb Classifiers

[SLIDE]

The verb classifiers in Slave present a good example of a single stem from the morphosyntactic point of view having multiple entries for different morphophonological properties. Classifiers show up in the prefix position immediately preceding the verb root.

One morphosyntactic entry::many morphophonological entries

[SLIDE]

The choice of classifier is idiosyncratic for the verb root, although further classifiers can be prefixed with productive valence-changing effects. Rice notes that there are cases where a given verb root will allow multiple different classifiers, without any morphosyntactic consequences. That is, these verb roots are somewhat underspecified as to their verb class, and this is purely a matter of morphophonology.

One morphophonological entry::many morphosyntactic entries

[SLIDE]

The analogous situation the other way would arise with homophony or polysemy. If there are distinct word senses associated with roots with identical morphophonological properties, with or without different valence properties, they would motivate multiple morphosyntactic entries which could 'share' the same morphophonological entry.

Correlated morphophonological and morphosyntactic choices

[SLIDE]

Finally, there may be cases where two stems might share the same underlying form, but differ in both morphophonological and morphosyntactic properties. One example of this is the class of roots that Rice analyzes as being neither inherently verbal nor inherently nominal. They can combine with affixes in typically verbal and typically nominal patterns. Depending on the morphophonological context, they are interpreted syntactically and semantically as nouns or verbs. Stems such as these will be associated with multiple morphophonological classes (one verbal, one nominal, say) and likewise multiple morphosyntactic classes.

In the case of known stems, with an inflectionally rich language like Slave, it probably won't be necessary to explicitly mark the correlation between morphophonolgical and morphosyntactic combinatory potential. This is because the surface verbal prefixes, for example, get passed to the morphosyntactic component as abstract morphemes. These morphemes will express things like subject or object agreement, tense-aspect-mood, incorporated stems, etc. Only the verbal morphosyntactic entries will be compatible with the lexical rules which involve those abstract morphemes.

On the other hand, there are two cases where the category information gleaned from the morphophonological context can be helpful. The first is the case where the morphosyntactic categories are both compatible in general with the same abstract affixes, but a difference in word-sense, say, correlates with a difference in something like classifier selection. The second is when dealing with unknown forms. That is, when the morphophonological analyzer is run in guesser mode, stripping off known affixes and proposing stems for which there are no lexical entries, either morphophonological or morphosyntactic. In this case, the category information from the morphophonology can be used to guide the choice of which default morphosyntactic entry to posit.

Conclusions

[SLIDE]

To conclude briefly then, we have argued that for the purposes of grammar engineering for language documentation, morphophonology and morphosyntax are best treated as independent, interfacing systems. The point at which they interface is in abstract morphemes. Furthermore, two kinds of interface are required: On the one hand, the run-time system must have a way of passing information from the morphophonology to the morphosyntax and vice versa. On the other hand, during development, the linguist must be able to see the relationship between morphophonological and morphosyntactic entries. Finally, and not surprisingly, in designing a system of this type, morphologically exuberant language families like Athabaskan can be very informative.