Implementation for discovery: A bipartite lexicon to support morphological and syntactic analysis[1]

Emily M. Bender and Jeff Good
University of Washington and MPI for Evolutionary Anthropology

1 Introduction

The purpose of this paper is to present and justify aspects of the Montage model of morphology. Montage (Bender et al. 2004) is a long-term project with the goal of building a suite of software tools to assist linguists in the documentation of underdescribed languages by allowing them to make use of techniques from grammar engineering without becoming grammar engineers themselves.

An important aspect of the development of these tools is devising models of grammatical phenomena which are both computationally tractable and intuitive to the descriptive linguist. A particularly thorny instance of this is in the development of a sufficiently general model for morphological phenomena, since such phenomena can involve complex interactions among a number of aspects of a language's grammar. Accordingly, a major area of research within the Montage project, at present, is coming to an understanding of the descriptive linguist's process of morphological discovery in order to identify what aspects of that process can be facilitated by existing techniques from grammar engineering and to, thereby, develop a general model of morphological analysis to be supported by Montage tools.

This paper discusses Montage's findings to date in this area. §2 gives some relevant background information about the Montage project. §3 justifies the abstract model of morphology adopted by Montage. §4 discusses how we are implementing that model at present. And, §5 points out important areas of future work for this aspect of the Montage project.

2 Background: Montage

The research that is the topic of this paper is situated within the larger Montage project (Bender et al. 2004), which has the goal of building a toolkit to assist linguists in the documentation of underdescribed languages by facilitating text analysis, grammar development, and the creation of links between texts and an accompanying dictionary. The project is based on two key insights: (i) the field of grammar engineering has reached a state of maturity where its techniques can be used beyond its core domain of computational linguistics and be of use to field linguists as well and (ii) in recent years, field linguists have become aware of the need to develop rigorous standards for digital resources to ensure their longevity, and these standards will make those resources more amenable to being exploited by compu-

[1] We would like to thank Ann Copestake, Anya Dormer, David Goss-Grubbs, Mike Maxwell, Tony Woodbury, Scott Drellishak, Jeremy Kahn, Bill McNeill, Matty Noble, Laurie Poulson, audiences at the 2005 LSA and SSILA meetings in Oakland, California and at the 2005 meeting of CLS for helpful discussion.

tational techniques than they have previously been (see, for example, the work of the E-MELD project discussed by Moran (this volume)).

The guiding principles of the Montage project are:

- When possible, make use of existing standards and tools from grammar engineering (e.g., LKB (Copestake 2002), XFST (Beesley and Karttunen 2003), the Grammar Matrix (Bender et al. 2002)) and language documentation technology (e.g., XML standards for interlinear text (Bow et al. 2003), linguistic paradigms (Penton et al. 2004), and descriptive grammars (Good 2004)).
- Develop models of linguistic analysis which are computationally tractable without constraining possible analyses in ways which would be unnatural to the documentary linguist.
- Do not build tools which attempt to *do* grammatical analysis. Rather, identify steps of the analytical process which are amenable to automation and build tools which facilitate the work done during those steps.
- Incorporate best-practice standards and methodologies into tool-design to facilitate resource archiving and interoperability.

Montage is a long-term project, and it will take many years to develop the entire toolkit. Here, we focus on one important aspect of the project: The Montage model of morphological analysis and its implementation.

3 The Montage Model of Morphology

3.1 Introduction

In this section, we discuss two logically-possible abstract models for morphological analysis that could be adopted by Montage, but which we reject, en route to justifying our adoption of a third model. In the discussion, we assume the specific senses of the terms *morphophonology* and *morphosyntax* as defined in (1), in order to draw the distinctions which are important in this work.

(1) a. **Morphophonology:** The mapping of surface forms, represented as strings, to sets of abstract morphemes.

b. **Morphosyntax:** The mapping of sets of abstract morphemes to syntactic/semantic structures.

In linguistic terms, we can say that, within Montage, morphophonology is taken to include general (word-level) phonological processes, morphologically conditioned phonological rules, morphologically conditioned allomorphy, and the mapping between abstract morphemes (e.g., “nominative case”) and underlying phonological representations. Morphosyntax is taken to include the means for constructing full syntactic and semantic representations of inflected lexical items out of sets of abstract morphemes. Both morphophonology and morphosyntax may include a representation of morphotactics (morpheme ordering, morpheme co-occurrence restrictions).

As will be discussed in 3.4, the Montage model of morphology treats morphophonology and morphosyntax as largely independent from each other. However, given that analysis along both dimensions is necessary for fully parsing or generating linguistic structures, it is not immediately clear why, from the perspective of machine implementation, the two should not be treated in a more integrated way. In §3.2 and §3.3, we discuss two logically-possible models for morphophonological and morphosyntactic integration which we label *morphophonology in morphosyntax* and *morphosyntax in morphophonology*. We will see that neither model has the ability to handle the range of morphological phenomena which the Montage toolkit will need to support to be maximally useful to descriptive linguists.

In the discussion, we will draw on examples from the Athabaskan language Slave as discussed by Rice (1989). Montage has chosen Slave as a language on which to test its tools and models because of the notoriously complex morphosyntax of Athabaskan languages. We assume that a system that can do a reasonably good job of assisting with the analysis of an Athabaskan language will work well for the vast majority of underdescribed languages.

3.2 Morphophonology in Morphosyntax

By *morphophonology in morphosyntax*, we are referring to models where a given morphological construction, say, for example, a plural construction, are treated as being primarily morphosyntactic in nature accompanied by a morphophonological "side effect", perhaps affixation or a stem change. This model has been more or less standard in Head-driven Phrase Structure Grammar (see, e.g., Pollard and Sag 1994; Orgun 1996) and it is adopted in the English Resource Grammar (Flickinger 2000), as well as most if not all other grammars developed with the LKB grammar development environment (Copestake 2002).[2]

While a morphophonology in morphosyntax model is manageable for a language with relatively sparse morphology, like English, it is not well suited for languages with complex morphology for at least two reasons: (i) such a model makes it difficult to reuse morphophonological analyses when the same morphophonological rules apply in different morphosyntactic constructions and (ii) morphophonological rules with no morphosyntactic effects are difficult to formalize. In addition, we have found that morphophonology in morphosyntax, when combined with a rule-based (rather than paradigm-based) approach to morphology, makes the specification of morphophonological lexical idiosyncrasy overly cumbersome. We will illustrate the first two drawbacks of morphophonology in morphosyntax using examples from Slave. and the third with more general considerations.

[2] Note that morphosyntactic and morphophonological rules are formally distinct in the LKB, and thus could be in principle be treated as more separate in the grammars.

3.2.1 Same morphophonology; different morphosyntax

The *D-effect rule* found in Slave is a good illustration of the value of being able to reuse morphophonological analysis in different syntactic constructions. A basic characterization of the rule is that "[t]he *d-* classifier and the final /d/ of the first person plural morpheme *íd-* combine with an immediately following stem initial consonant" (Rice 1989:129).[3] Among other things, this rule causes a *d+ʔ* combination to be realized as *t'* (an ejective) as in the examples in (2). For some consonants, the D-effect rule causes the *d* to be deleted entirely.[4]

(2) a. *ya-de-**d**-Ø-**ʔ**áh* ⟶ *yádeh**t'**ǫ*
ADV-INC-D-Ø-be.fooled
"I was fooled." (Rice 1989:444)

b. *íd-Ø-ʔáh* ⟶ *yí**t'**ah*
1PL-Ø-go
"We two are going." (Rice 1989:476)

The D-effect rule is not strictly phonological in nature. This can be seen by examining morpheme-internal sequences of *dh*, exemplified in (3a) versus sequences derived by the appearance of *íd-* first-person plural morpheme before an *h* classifier, exemplified in (3b). In (3a) both consonants in the *dh* sequence surface, in (3b), neither consonant surfaces, with the *d* deleted as part of the D-effect rule and the *h* deleted as the result of another process.

(3) a. *ta-**dh**ee* ⟶ *ta**dh**ee*
water-hot
"open water" (Rice 1989:185)

b. ***íd-h**-t'ó* ⟶ *yí**Ø**t'ó*
1PL-H-suck
"we sucked" (Rice 1989:477)

If morphophonological effects were treated as directly accompanying specific morphosyntactic constructions, the phonological alternations associated with the *d*-classifier and the *íd-* first-person plural prefix would have to be stated twice, once for each of these two morphemes. From an analytical perspective, this would amount to missing a generalization. From the perspective of a documentary linguist using a computational tool in their work, it would not only mean duplicating their

[3] Within Athabaskan linguistics, the term *classifier* refers to a small set of thematic/voice prefixes which appear immediately preceding the verb stem (Rice 1989:437).

[4] The glossing abbreviations used in this paper are as follows: ADV 'adverb', INC 'inceptive', REP 'repetitive', D '*d*-classifier', Ø 'null classifier', H '*h*-classifier'. In some of the examples, the morphological parsing is not given in Rice (1989) and is our own. These parsings may not be complete, especially with respect to the identification of null morphemes.

effort but would also, in all likelihood, mean forcing them to implement an analysis they would find unnatural. Furthermore, such an implementation would be error prone, leaving open the possibility that the linguist would revise the rule in one case but forget about the other. Therefore, from both an analytical and a user-oriented perspective, it appears that the morphophonology in morphosyntax model is not appropriate for Montage.

3.2.2 Morphosyntactically irrelevant morphophonology

Another class of morphophonological phenomena which argues against the morphophonology in morphosyntax model are cases where generalizations about morphological structure are not associated with morphosyntactic information. An example of such a phenomenon can be found in Rice's (1989) analysis of the insertion of a prefixal peg-element, with form *he-*, in Slave verbs when there would not otherwise be a syllable preceding the verb stem. The appearance of this element can be seen in (4a), where it is bolded. The verb in (4a) can be contrasted with the one in (4b), which is based on the same stem but marked with the second person singular prefix which, unlike the null third singular prefix, allows the verb stem to be preceded by a syllable without insertion of the peg element.

(4) a. *Ø-jį* ⟶ ***he**jį*
3SG-sing
"he or she sings" (Rice 1989:133)

b. *ne-jį* ⟶ ***ne**jį*
2SG-sing
"you (sg.) sing" (Rice 1989:133)

To the extent that the peg element in Slave cannot be associated with any meaning, grammatical or otherwise, accounting for it within a morphophonology in morphosyntax model would not be impossible but would require a somewhat awkward analysis involving defining a syntactically and semantically "null" morphosyntactic construction which would only apply if the right phonological conditions were met—clearly an undesirable way to formalize or implement Rice's (1989) analysis and another reason to not adopt that model within Montage.

Of course, another way to deal with phenomena like the Slave peg element under a morphophonology in morphosyntax approach would be to suggest that the peg element is not, in fact, meaningless and to determine what meaning it could have. Its appearance could then be analyzed straightforwardly as being part of the exponence of the relevant morphosyntactic construction. In fact, Hargus and Tuttle (1997) do propose that peg elements in Athabaskan languages can be associated with some meaning—namely, that they can be analyzed as a type of tense/aspect prefix.

From the perspective of Montage, however, the fact that such a phenomenon *can* be analyzed as a morphosyntactic element in addition to a morphophonological one is far less important than the fact that it *has* been analyzed as purely morphophonological in nature. As discussed in §2, the goal of Montage is to facilitate linguistic analysis, while imposing minimal restrictions on what analyses are supported. Thus, to the extent that the morphophonological analysis of the Slave peg element is a reasonable one, we would want to adopt a morphological model that would allow it to be formalized and implemented in a straightforward way.[5]

3.2.3 Interaction with rule-based morphology

In the Montage system to date, we have adopted a rule-based approach to morphology as opposed to a paradigm-based approach (such as, e.g., Krieger and Nerbonne 1993). In a rule-based, morphophonology in morphosyntax approach, we find ourselves specifying morphophonological lexical idiosyncrasies in the definition of the rule, rather than in the definition of the lexical entries for the particular words involved. We would prefer a system in which such facts as irregular stems (e.g., French *ir-* in the future tense of *aller* 'to go') and irregular affixes selected by particular stems (e.g., English *child/children/*childs*) are consolidated as morphophonological facts about particular lexical items. While it should be possible in principle to construct a morphophonology in morphosyntax system which nonetheless obeys this constraint, we find it more natural in a system with independent morphophonology and morphosyntax. We note that in primary documentary linguistic work, linguists often begin by working with all morphophonological facts as if they were lexically specific, and then search for generalizations across the words collected. Thus for incremental development, an elegant, user-friendly treatment of lexically-specific facts is crucial.

Having discussed ways in which a morphophonology in morphosyntax analysis is inadequate for Montage, in the next section we discuss another possibility: morphosyntax in morphophonology.

3.3 Morphosyntax in Morphophonology

Another logical possibility for a morphological analysis system would be to attempt to do morphosyntactic analysis using a parsing/generating system primarily designed for morphophonology. This would amount to representing "underlying" morphemes in a way which would allow them to be interpreted as abstract feature bundles. Consider, for example, the two possible "parses" of the word *cats* in (5).

(5) a. cat-plural

b. [SYNTAX noun, SEMANTICS cat] + [SEMANTICS plural]

While we are aware of no large-scale implementation of a morphosyntax in morphophonology model, it has been proposed, at least, on a small scale. Beesley and

[5] In fact, in the case of this particular phenomenon, while Hargus and Tuttle (1997:193) present evidence against a purely morphophonological analysis of the peg element from a Pan-Athabaskan perspective, they do suggest that Rice's analysis is adequate for Slave.

Karttunen (2003:343–349), for example, illustrate how the flag-diacritic features of the XFST finite-state transducer can be used to do some rudimentary morphosyntactic analysis, using the example of Arabic definite and indefinite case endings. This kind of analysis is plausible when the morphosyntactic side involves merely adding or “filling-in” values of particular features (case, definiteness). It doesn’t scale, however, to more complex morphosyntactic phenomena such as voice alternations or category-changing derivational processes.

Morphological causatives, for example, are morphosyntactically too complex to be processed by a device, like a finite-state transducer, designed primarily for morphophonological analysis. For illustrative purposes, a non-causative/causative sentence pair from Slave is given in (6).

(6) a. *ʔelá k’e-Ø-leh*
boat REP-Ø-float
“the boat is floating”
(Rice 1989:455)

b. *ʔelá k’e-h-Ø-leh*
boat REP-H-Ø-float
“he or she is floating the boat”
(Rice 1989:455)

Sentence (6a) is headed by a non-causative verb. Sentence (6b) is headed by the causative variant of the same verb. The causative verb is formally marked by the addition of an *h*-classifier to the verb (which attaches to the left of a null classifier).

Unlike, say, a plural morpheme, the morphosyntactic effects of causatives cannot be expressed through the simple addition of a feature like “CAUS”. The differences between the morphosyntactic properties of a non-causative verb and its causative counterpart include, at least, the addition of causative semantics and also a *change* in the argument structure of the verb to allow an additional causer argument to be expressed. In feature structure notation, the non-causative variant of the verb ‘float’ is represented in Figure 1a, while the causative variant is represented in Figure 1b.

$$\text{a.}\quad \begin{bmatrix} \text{ARG-ST} & \langle\, \mathrm{NP}_i \,\rangle \\ \text{SEM.RELATIONS} & \langle\, \mathrm{float}(e,i) \,\rangle \end{bmatrix} \qquad \text{b.}\quad \begin{bmatrix} \text{ARG-ST} & \langle\, \mathrm{NP}_j, \mathrm{NP}_i \,\rangle \\ \text{SEM.RELATIONS} & \left\langle \begin{array}{l} \mathrm{cause}(e2, j, e1), \\ \mathrm{float}(e1,i) \end{array} \right\rangle \end{bmatrix}$$

Figure 1: Feature structure representation of argument structure and semantics of non-causative and causative *float*

In order to be able to state the relationship between Figure 1a and Figure 1b in a general way, the formal system used for the morphophonology would need to be far more powerful than seems to be otherwise required.[6]

Our conclusion is, therefore, that neither morphophonology in morphosyntax nor morphosyntax in morphophonology are appropriate models of morphological analysis for a computational system like Montage. In the next section, we discuss the model of morphological analysis adopted by the Montage project: independent morphophonology and morphosyntax.

3.4 Independent Morphophonology and Morphosyntax

Having argued that neither a morphophonology in morphosyntax model nor a morphosyntax in morphophonology model is appropriate for Montage, we are left with the idea that morphophonology and morphosyntax are sufficiently distinct that they should be handled independently from each other. From a theoretical perspective, such a move is not unprecedented—one articulation of such a view can be found in Woodbury (1996). By adopting such a view Montage is not so much making a theoretical claim as a methodological one: Our present understanding of morphological phenomena is such that a toolkit which is sufficiently flexible to be of use in the description of *any* language should employ a model where morphophonological and morphosyntactic generalizations can be made largely independent of one another.

However, morphophonology and morphosyntax cannot be treated as *completely* independent from each other, since such a design would make it impossible for morphophonological and morphosyntactic analyses to interact in any meaningful way. Therefore, we propose a minimal interface between the two through the use of a bipartite lexical database where lexical entries can have distinct morphophonological and morphosyntactic components with a lexical ID serving to link the two.

We are certainly not the first to adopt a system where morphophonology and morphosyntax are handled as separate systems. Other such systems include the work of Ritchie et al. (1992), Antworth (1994), Aduriz et al. (2000), and Kaplan et al. (2004). The main contributions of the present work are (i) to motivate this kind of design in the context of linguistic documentation and discovery and (ii) to propose a means of keeping morphophonology and morphosyntax separate while allowing the linguist to build one integrated lexicon.

3.5 Incremental discovery in morphology

While Montage is not the first project to make use of an independent morphophonology and morphosyntax model, to the best of our knowledge, it is the first such project which is making use of such a model for a tool designed to aid linguists

[6] Bird and Klein (1994) have proposed a model for phonology which draws on the devices of typed feature structures, the same formalism used for syntax and semantics in HPSG. Such a system for morphophonology would clearly be sufficient for morphosyntactic generalizations as well. However, there is still considerable independence between morphophonology and morphosyntax: individual types can express either type of pattern, or correlate the two. Furthermore, they envision being able to implement phonological analysis in their system using finite-state techniques (p. 460).

is primary linguistic analysis and discovery. This can be contrasted with, for example, the systems discussed by Kaplan et al. (2004), Siegel and Bender (2002), which were designed for languages whose morphological systems were already well-studied (e.g., English or Japanese). In these, and many other such systems, the morphophonological analyzers are in fact built by different groups than the syntactic/semantic analyzers, and the former are treated as a "black box" by the latter. This is an option because the languages are well understood and have many people working on them.

However, "black box" approaches to morphophonology are inappropriate for Montage for two reasons: (i) In most cases, a single descriptive linguist is working on both morphophonological and morphosyntactic analysis. Asking him/her to maintain entries for each word in two separate systems is inefficient and error prone (as when data gets updated in one but not the other). (ii) From the perspective of a descriptive linguist, morphophonological analysis does not necessarily take place separate from morphosyntactic analysis, but, rather, in parallel with it, and a tool which is intended to play a role in morphological discovery must be able to support incremental analysis of a lexical item's morphophonological and morphosyntactic behavior. The use of a bipartite lexicon in the Montage system gives it the needed flexibility to allow for such incremental analysis.

As we will see in §4, the morphophonology independent from morphosyntax model is built into the design of the morphological analysis system in two distinct, but related, ways. From a parsing/generation perspective, each class of phenomena is handled by a separate analyzer. From a development perspective, each class occupies a different "slot" in a morpheme's lexical entry, thus creating a system which can simultaneously support incremental analysis as well as parsing and generation based on whatever analysis has been encoded at a given time.

In the next section, we discuss the proposed Montage implementation of the system for morphological analysis just discussed.

4 The Implementation

4.1 Introduction

Having discussed why the Montage project has adopted a morphological model where morphophonology and morphosyntax are treated as largely independent, in this section, we discuss how we propose to implement this model within a sentence parsing and generation system using a bipartite lexical database. In §4.2 we discuss the structure of the bipartite lexical database. In §4.3 we discuss how the bipartite lexical database fits into a parsing and generation system, and in §4.4 we discuss some of the methods we have developed for building necessary implemented grammar resources from the database for use in the parsing and generation system.

4.2 A Bipartite Lexical Database

As discussed above, even though we are treating morphophonology and morphosyntax as essentially independent from each other, in order to provide integrated morphophonological and morphosyntactic analysis in a computational tool, they have to be interfaced in some way. Such an interface can be handled by the making use of a bipartite lexical database, as schematized in Figure 2, where separate morphophonological and morphosyntactic "slots" are linked by a single lexical ID.

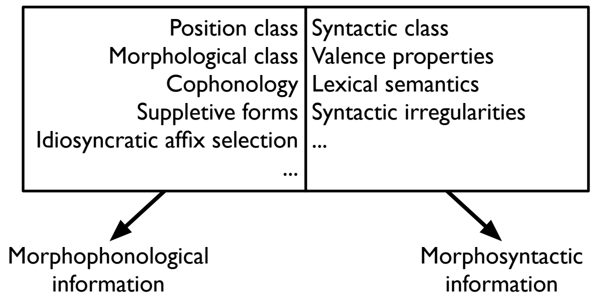

Figure 2: Bipartite lexical database entry

This database structure allows for morphophonological information to be separated from morphosyntactic information, thereby supporting independent morphophonological and morphosyntactic analysis and indeed allowing for many-to-many mappings between the two (multiple morphosyntactic entries using the same morphophonological entry, a single morphosyntactic entry with multiple morphophonological realizations). As will be seen in §4.3, this database structure additionally creates enough of an interface between morphophonology and morphosyntax to build an integrated parsing and generating system for morphological analysis.[7]

In using such a database to develop linguistic resources, we envision a process wherein a descriptive linguist incrementally updates the morphophonological and morphosyntactic components of a given lexical entry to reflect their knowledge about the behavior of a lexical item. The bipartite design of the database will allow them to update either half of the database without being concerned about the content of the other half.

In order to develop this database, we will be building on the PostgreSQL relational lexical database incorporated in the LKB (Copestake et al. 2004). An entry in the current LKB LexDB includes fields for a lexical identifier, a morphosyntactic type, an orthographic form, a semantic predication, and various bookkeeping information. In our proposed extension, the orthographic form would be replaced with a link to a set of information for morphophonological entry, including an underlying orthographic/phonological form, the string used to represent the abstract morpheme

[7] In addition, we anticipate that it will be useful to view lexical entries as having a third component containing lexicographic information, for example a morpheme's citation form and gloss.

(which may or may not be the same as the underlying form), position class information, co-phonology information, morphologically conditioned allomorphs, etc.

We should be quick to point out that, while the database will contain morphophonological and morphosyntactic information about lexical items, it will not contain all information necessary for parsing and generation of data. On the morphophonological side, phonological rules, for example, will have to be stated elsewhere as will information about natural classes and position classes. Similarly, on the morphosyntactic side, the syntactic constructions of the language will need to be defined in some other resource. Some of the additional resources needed for morphophonological analysis in the current implementation will be discussed in §4.4.

Finally, it bears mentioning that much work remains to be done in determining the full range of information which will need to be encoded in lexical entries of the database and how those lexical entries will be related to each other. For example, an important open research question is how best to represent paradigms within the Montage lexical database.

4.3 Runtime Interface

The role of the bipartite lexicon within a runtime interface based on an independent morphophonology and morphosyntax model for sentence analysis is schematized in Figure 3. For illustrative purposes, the diagram includes how the Slave string in (6b) would be parsed morphophonologically.[8]

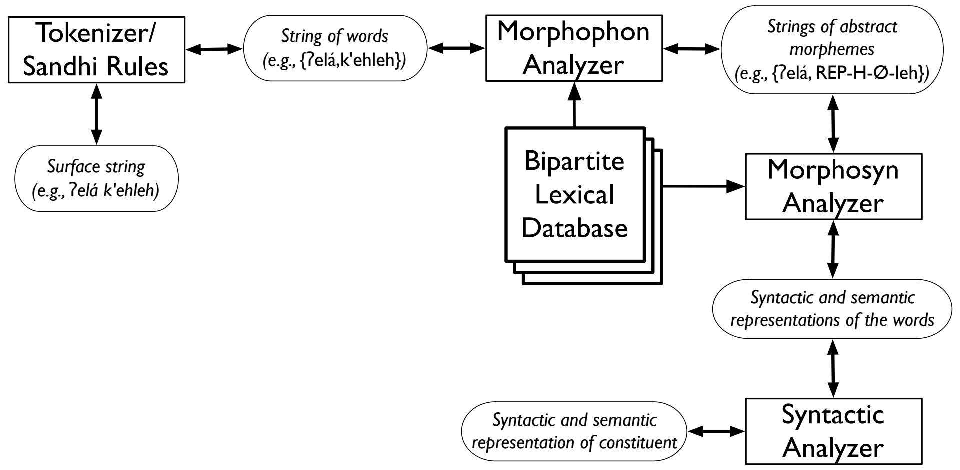
Tokenizer/ Sandhi Rules
String of words (e.g., {ʔelá,k'ehleh})
Morphophon Analyzer
Strings of abstract morphemes (e.g., {ʔelá, REP-H-Ø-leh})
Surface string (e.g., ʔelá k'ehleh)
Bipartite Lexical Database
Morphosyn Analyzer
Syntactic and semantic representations of the words
Syntactic Analyzer
Syntactic and semantic representation of constituent

Figure 3: Runtime interface using bipartite lexical database

As can be seen in the diagram, the bipartite database contributes information both to a morphophonological analyzer and a morphosyntactic analyzer and, thereby, plays

[8] The syntactic representations used by the syntactic analyzer are too complex to be straightforwardly included in Figure 3.

a key role in the interface between the two.[9] The model in Figure 3 is presently being approximated using the typed feature structure grammars developed within the LKB (Copestake 1992) for both the morphosyntactic and syntactic analyzer and an XFST (Beesley and Karttunen 2003) implementation of morphotactics and morphophonological rules.[10]

Not clearly represented in the diagram is the fact that during parsing or generation a given analyzer may have more than one output. For example, a given word may have more than one possible morphophonological parse or an optional phonological rule might cause two phonological outputs to be produced from the same abstract morphophonological representation.

4.4 From morphophonological rule to finite state transducer

LKB already provides a well-defined system for defining a database specifying the morphosyntactic aspects of each lexical item. In addition, the Grammar Matrix (Bender et al. 2002, Bender and Flickinger 2005) promises to make the grammar engineering work of relating morphosyntactic properties (e.g., case, argument structure) of lexical items to their syntactic and semantic behavior relatively approachable for non-grammar engineers. However, the task of converting structured morphophonological information about a lexical item into a representation which can be used by XFST has not been well-explored. Therefore, we discuss our present implementation of this aspect of our system in more detail here.

In order to automate the process through which information in the morphophonological half of the lexical database can be transformed into a format usable by XFST (in particular, for a fragment of a grammar of Slave), we have had to make use of several resources containing information not appropriate for the database itself:

- **Morphophonological rule definitions:** A set of morphophonological rules, writable in a form expected to be familiar to a descriptive linguist, which can be classified according to linguist-defined types. These include both phonological rules with morphological conditioning and purely phonological rules.
- **Character class definitions:** A set of definitions of character classes so that an abbreviation (e.g., C) to stand in for a set of segments (e.g., consonants).
- **Position class definitions:** A set of definitions for linguist-determined morphological position classes. Includes information like a name for the position class and whether or not it is an optional or obligatory position.

[9] In the present design, the morphosyntactic analyzer is permitted to access one type of morphophonological information: an abstract form of the morphophonology of a morpheme. This allows the morphosyntactic analyzer to “look up” the morphosyntactic information of the morphemes outputted by the morphophonological analyzer.

[10] In the Slave examples we are using to test the system, all words have been indicated with spaces and no sandhi rules have been represented in the orthography. Therefore, the default white-space tokenization provided by the LKB has been sufficient.

In the present implementation, the morphophonological half of a lexical item's entry contains the following information: (i) an underlying phonological form, which serves as the input to the morphophonological and phonological rules, (ii) an "abstract morpheme" representation, either the same as the underlying phonological form or something more like a gloss, for grammatical morphemes, (iii) a position class name, if relevant, (iv) any cooccurrence restrictions, if relevant (i.e., a list of IDs of lexical items this morpheme must appear with), (v) a list of phonological rule classes for the morpheme, (vi) for stems, a list of stem alternations and their conditioning environments and any irregular affix forms selected by the stem.[11]

All of this information is extracted from the various resources and used to create two XFST files, one corresponding to a lexicon (in the form of an XFST lexc file) and the other corresponding to a set of phonological rules (in the form of an XFST script file). The somewhat complex process is schematized in Figure 4, which shows how information from a number of resources contributes to the development of the XFST files, a lexc file labeled Morph Lexicon and a script file labeled Morph Rules. Once those files are created, they can be read by XFST to produce a finite state transducer which can parse surfacing word forms to produce a sequence of abstract morphological representations or generate surface forms given a sequence of abstract morphemes.

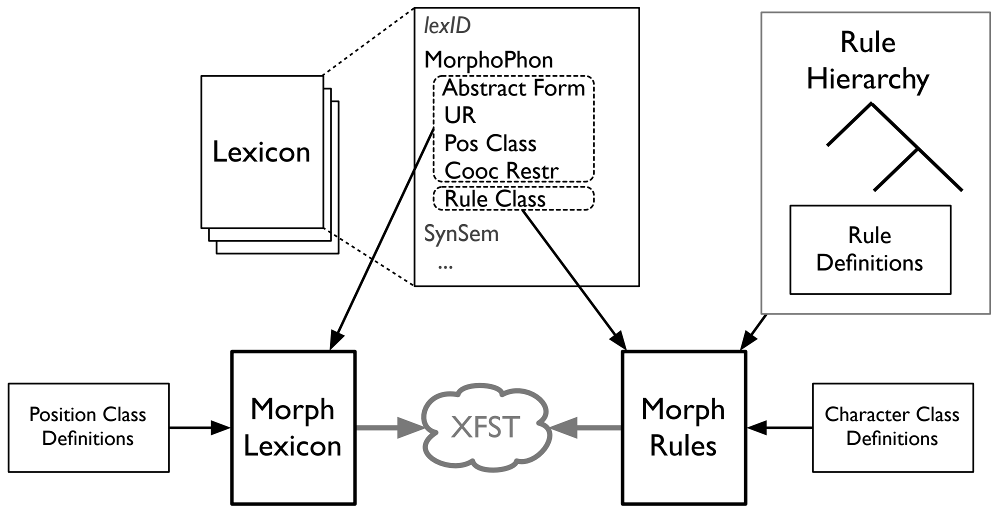

Figure 4: Creation of XFST resources from lexical database

As shown in Figure 3, the output of the morphophonological analyzer is a string

[11] Bird and Klein (1994) have suggested that it would be preferable, on theoretical and practical grounds, to formalize phonological generalizations in implemented grammars using just one level of phonological representation (and, thereby, not making use of notions like "underlying form"). Montage, at present, is assuming a system design using multiple levels of phonological representation since we believe that will be more familiar and intuitive to the typical field linguist. Future research on Montage will help us determine if this is an appropriate assumption.

of abstract morphemes (or a set of strings of morphemes, in the case of ambiguous forms). This "normalized" string is then the input to the morphosyntactic/syntactic analyzer which uses it to build a complete syntactico-semantic representation of each word. Note that this second step involves quite a bit of what is traditionally though of as "morphology", but no morphophonology. In some cases, a form may be morphophonologically legitimate, but morphosyntactically ill-formed. In these cases, it goes "through" the morphophonological analyzer but is blocked at the next stage. In the case of ambiguous inputs, the morphosyntactic analyzer builds multiple representations for each word, filtering out representations which are morphosyntactically ill-formed.

5 Future work

Development of the Montage toolkit has just begun, and there is clearly much work to do on the morphological system as well as other aspects. In addition to implementing the database extension described in §4.2, this work will include: (i) developing an intuitive user-interface for descriptive linguists; (ii) exploring facilities to support testing of possible rule interactions (feeding and bleeding relationships) and the appropriate "order" in which to apply the rules; (iii) exploring means of starting from underlying form-surface form mappings and generating hypothesized phonological rules; (iv) support for sandhi/phrase level phonology; and (v) exploring the division of labor between morphophonology and morphosyntax in ruling out illegitimate forms.

References

Aduriz, I., E. Agirre, I. Aldezabal, I. Alegria, X. Arregi, J. M. Arriola, X. Artola, K. Gojenola, A. Maritxalar, K. Sarasola, and M. Urkia. 2000. A word-grammar based morphological analyzer for agglutinative languages. In *Proceedings of the 17th conference on Computational linguistics*, 1–7, Morristown, NJ, USA. Association for Computational Linguistics.

Antworth, Evan L. 1994. Morphological parsing with a unification-based word grammar. In *North Texas Natural Language Processing Workshop, May 23, 1994, Arlington, Texas*.

Beesley, Kenneth R., and Lauri Karttunen. 2003. *Finite State Morphology*. Stanford: CSLI.

Bender, Emily M., and Dan Flickinger. 2005. Rapid prototyping of scalable grammars: Towards modularity in extensions to a language-independent core. In *Proceedings of the 2nd International Joint Conference on Natural Language Processing IJCNLP-05*, Jeju Island, Korea.

Bender, Emily M., Dan Flickinger, Jeff Good, and Ivan A. Sag. 2004. Montage: Leveraging advances in grammar engineering, linguistic ontologies, and mark-up for the documentation of underdescribed languages. In *Proceedings of the Workshop on First Steps for Language Documentation of Minority Languages: Computational Linguistic Tools for Morphology, Lexicon and Corpus Compilation, LREC 2004, Lisbon, Portugal*.

Bender, Emily M., Dan Flickinger, and Stephan Oepen. 2002. The Grammar Matrix: An open-source starter-kit for the rapid development of cross-linguistically consistent broad-coverage precision grammars. In *Proceedings of the Workshop on Grammar Engineering and Evaluation at the 19th Internation Conference on Computational Linguistics*, 8–14, Taipei, Taiwan.

Bird, Steven, and Ewan Klein. 1994. Phonological analysis in typed feature systems. *Computational Linguistics* 20(3):455–491.

Bow, Cathy, Baden Hughes, and Steven Bird. 2003. Towards a general model for interlinear text. In *Proceedings of the E-MELD Workshop 2003: Digitizing and annotating texts and field recordings, East Lansing, Michigan, July 11–13*.

Copestake, Ann. 1992. The ACQUILEX LKB: Representation issues in the semi-automatic acquisition of large lexicons (ACQUILEX WP No. 36). In A. Sanfilippo (Ed.), *The (other) Cambridge ACQUILEX papers*. University of Cambridge Computer Laboratory, Technical report No. 253.

Copestake, Ann. 2002. *Implementing Typed Feature Structure Grammars*. Stanford, CA: CSLI Publications.

Copestake, Ann, Fabre Lambeau, Benjamin Waldron, Francis Bond, Dan Flickinger, and Stephan Oepen. 2004. A lexical module for a grammar development environment. In *Proceedings of LREC 2004*.

Flickinger, Dan. 2000. On building a more efficient grammar by exploiting types. *Natural Language Engineering* 6 (1) (Special Issue on Efficient Processing with HPSG):15 – 28.

Good, Jeff. 2004. The descriptive grammar as a (meta)database. In *Proceedings of the E-MELD Workshop 2004: Linguistic databases and best practice, July 15–18 2004, Detroit, Michigan*.

Hargus, Sharon, and Siri G. Tuttle. 1997. Augmentation as affixation in Athabaskan languages. *Phonology* 14:177–220.

Kaplan, Ronald M., John T. Maxwell III, Tracy Holloway King, and Richard Crouch. 2004. Integrating finite-state technology with deep LFG grammars. In *Proceedings of the Workshop on Combining Shallow and Deep Processing for NLP* (ESSLLI 2004), Nancy, France.

Krieger, Hans-Ulrich, and John Nerbonne. 1993. Feature-based inheritance networks for computational lexicons. In T. Briscoe, A. Copestake, and de Paiva Valeria (Eds.), *Inheritance, Defaults, and the Lexicon*, 90–136. New York: Cambridge University Press.

Moran, Steven. This volume. The E-MELD School of Best Practices in digital language documentation. In *Proceedings of the 41st Annual Meeting of the Chicago Linguistic Society*.

Orgun, C. Orhan. 1996. *Sign-based morphology and phonology with special attention to Optimality Theory*. PhD thesis, UC Berkeley. Available as ROA 171-0197.

Penton, David, Catherine Bow, Steven Bird, and Baden Hughes. 2004. Towards a general model for linguistic paradigms. In *Proceedings of the E-MELD Workshop 2004: Linguistic databases and best practice, July 15–18 2004, Detroit, Michigan*.

Pollard, Carl, and Ivan A. Sag. 1994. *Head-driven Phrase Structure Grammar*. Chicago: Chicago University Press.

Rice, Keren. 1989. *A Grammar of Slave*. New York: Mouton de Gruyter.

Ritchie, Graeme D., Graham J. Russell, Alan W. Black, and Steven G. Pulman. 1992. *Computational Morphology: Practical Mechanisms for the English Lexicon*. Cambridge, MA: The MIT Press.

Siegel, Melanie, and Emily M. Bender. 2002. Efficient deep processing of Japanese. In *Proceedings of the 3rd Workshop on Asian Language Resources and International Standardization. COLING 2002 Post-Conference Workshop, August 31*, Taipei, Taiwan.

Woodbury, Anthony C. 1996. On restricting the role of morphology in Autolexical syntax. In E. Schiller, E. Steinberg, and B. Need (Eds.), *Autolexical theory: Ideas and methods*, 318–363. Berlin:Mouton.