A High-level Morphological Description Language Exploiting Inflectional Paradigms

Peter Anick and Suzanne Artemieff

Digital Equipment Corporation
111 Locke Drive, LMO2-1/D12
Marlboro, MA 01752
anick@aiag.enet.dec.com

Abstract

A high-level language for the description of inflectional morphology is presented, in which the organization of word formation rules into an inheritance hierarchy of paradigms allows for a natural encoding of the kinds of rules typically presented in grammar books. We show how the language, composed of orthographic rules, word formation rules, and paradigm inheritance, can be compiled into a run-time data structure for efficient morphological analysis and generation with a dynamic secondary storage lexicon.

1 Introduction

Pedagogical grammar books typically organize their descriptions of the inflectional morphology of a language in terms of paradigms, groups of rules which characterize the inflectional behavior of some subset of the language's vocabulary. A French grammar may divide verbs into the first, second, and third conjugations; German grammars speak of "weak" and "strong" verbs; Spanish grammars classify verbs by their infinitival endings, etc. The family of word forms that each vocabulary item may have can thus be described by a combination of a base stem (such as the "citation form" used to index words in a dictionary) and the paradigm the word belongs to. Irregular words, which exhibit behaviors not completely captured by general paradigms, often tend to be partially describable by reference to regular paradigmatic patterns.

The word formation rules that comprise a paradigm are usually expressed in terms of a sequence of stem change and affixation operations. For example, one French textbook [NEBEL74], in describing first conjugation verbs, shows how to form present tense forms using the infinitival stem with its "er" suffix removed. Future tense is formed by appending affixes to the full infinitival stem, while the stem of the imperfect tense is found by taking the first person plural of the present tense and dropping the "ons". In addition to such word formation rules, there are spelling change rules which describe variations in spelling, often conditioned by the phonological or orthographic context in which a word formation rule is applied.

While the above characterization of morphological behavior is a familiar one, most description languages that have been developed for computational morphology (e.g., [KOSKENNIEMI84], [GÖRZ88]) have tended to focus more on the orthographic and affixation rules, and pay less attention to explicitly capturing the regularities within and between paradigms. Recently, some researchers have begun exploring the advantages to be derived from a notation in which paradigms play a more central role (e.g., [CALDER89], [RUSSELL91]). This paper presents such a notation, called PDL (for Paradigm Description Language), which we are using as the basis of the morphological analyzer for AI-STARS, a multi-lingual "lexicon-assisted" information retrieval system ([ANICK90]). It has been a goal of our high-level language design to preserve, as much as possible, the kinds of descriptive devices traditionally used in grammar books.

Our approach to the representation of paradigms borrows from the Artificial Intelligence community's notion of "frames", data structures made up of slots with attached procedures, organized hierarchically to support default slot inheritance and overrides (e.g., [BOBROW77]). In a paradigm's "frame", the slots correspond to surface and stem forms, whose values are either explicitly stored (in the lexicon) or else computed by word formation rules. The hierarchical organization of paradigms helps to capture the shared linguistic behaviors among classes of words in an explicit and concise manner.

Our application domain introduces several constraints on the design of its morphological component:

- The morphological recognizer must work with a dynamic secondary storage lexicon accessed via an index on stem forms. This constraint rules out approaches relying on a left to right scan of the word using special in-memory letter tree encodings of the dictionary (e.g., [GÖRZ88]). It requires an approach in which potential stems are derived by affix removal/addition and/or stem changes and then probed for in the lexicon.

- The morphological information must additionally support surface form generation and "guessing". The guesser, to be employed in computer-assisted lexicon acquisition, must be able to construct potential *citation* forms (e.g., infinitive forms for verbs), not just stripped stems.
- The high-level language (PDL) must be compilable into a form suitable for efficient run-time performance. This implies not only efficient in-memory data structures but also a system which minimizes disk (lexicon) accesses.

Our aim is to develop morphological representations for a number of (primarily European) languages. We have built fairly complete representations for English, French, and German, and have begun investigating Spanish. While it is premature to predict how well our approach will apply across the range of European languages, we have found it contains a number of desirable aspects for applications such as AI-STARS.

In the next section, we provide an overview of the PDL language, describing how word formation rules are organized into a hierarchy of paradigms and how the lexicon and morphological rules interact. Then we provide an illustration of the use of paradigm inheritance to construct a concise encoding of French verb forms. Next we present algorithms for the compilation of PDL into efficient runtime data structures, and for the recognition and generation of word forms. We conclude with an evaluation of the strengths and weaknesses of the approach, and areas for future research.

2 Paradigm Description Language

Our paradigm description language (PDL) is composed of three major components - form rules, an inheritance hierarchy of paradigms, and orthographic rules.

2.1 Form Rules

We divide word forms into

- *surface forms*, which are those that show up in a text,
- *lexical forms*, which are those that are stored directly in the lexicon, and
- *intermediate forms*, those forms created by affixation or stem-change operations applied to other forms. These forms may not ever show up in a text but are useful in describing intermediate steps in the construction of surface forms from lexical forms.

In the form construction rules, we distinguish between two major categories of strings. *Stems* are any forms which include the primary lexical base of the word, whereas *affixes* comprise the prefixes and suffixes which can be concatenated with a stem in the process of word formation. Once an affix is appended to or removed from a stem, the result is also a stem, since the result also includes the primary lexical base. Form construction rules are restricted to the five cases below:

- <form> : <stem> + <affix>
- <form> : <stem> - <affix>
- <form> : + <affix> <stem>
- <form>: - <affix> <stem>
- <form> : <stem>

The <form> is a name for the string form created by the rule. <stem> is the name of a stem form. <affix> may be a prefix or suffix string (or string variable), its type (i.e., prefix or suffix) implied by its position before or after the <stem> in the rule. The operator (+ or -) always precedes the affix. If +, then the affix is appended to the stem as a prefix or suffix. If -, then the affix is removed from the stem. The resulting <form> name may in turn be used as a stem in the construction of some other form. In this way, the construction of a surface form may be described via a succession of affixation or stem-change operations, each operation described in a single rule.

The special symbol LEX may be used in the right-hand-side of a form rule to indicate that the form is stored as a lexical stem in the lexicon.

Grammatical features may be associated with form names, as follows:

```
<form>  [<feature> = <value>,
          <feature> = <value>, ...]
```

2.2 Paradigms

A paradigm in PDL is composed of a set of form construction rules which collectively characterize the family of surface forms for those words which belong to that paradigm. To capture the similarities among paradigms and to avoid redundancy in the description of a language, we allow one paradigm to be *based on* another paradigm. If paradigm B is based on paradigm A, then all the forms and form construction rules that have been defined for paradigm A also apply, by default, to paradigm B. We can then differentiate paradigm B from A in three ways:

1. We can add new forms and their construction rules for forms that do not exist in A.

2. We can rewrite (override) the construction rules for forms that do exist in A.

3. If a form in A is no longer applicable in B, we can delete it from B.

Note that the feature set(s) associated with form names cannot change from paradigm to paradigm; form names are universal, denoting the same features regardless of where they appear.

In order to facilitate the capture of generalizations across paradigms, we allow the definition of abstract paradigms. These are paradigms to which no words of a language actually belong, but which contain a set of forms and constructions which other paradigms have in common. Thus a concrete paradigm may be based on some other concrete paradigm or on an abstract paradigm. Likewise, an abstract paradigm may itself be based on yet another abstract (or concrete) paradigm.

The ability to base one paradigm on another, combined with the ability to represent intermediate stem forms as slots in a paradigm, is a very powerful feature of our morphological description language. Not only does it allow for paradigm descriptions that correspond closely with the kinds of descriptions found in grammar books, but, since the regularities among paradigms can be abstracted out and shared by multiple paradigms, it allows for very concise descriptions of inflectional behavior (including subregularities often overlooked in grammar books), as illustrated in section 3.

2.3 Orthographic Rules

Form construction rules describe which stems can combine with which affixes to create new forms. The concatenation or removal of an affix may in some cases result in a spelling change other than the mere concatenation or removal of the affix string. In English, many words ending in a vowel followed by a consonant will double the final consonant when an affix starting with a vowel is appended. In French, the addition of certain affixes requires that an "e" in the stem of some verbs be rewritten as "è". Since these spelling change rules are often based on general phonological/orthographic properties of affixes and stems, rather than the specific form rules themselves, and hence may apply across paradigms, we support the independent specification of spelling rules capturing these changes. Each rule is written to apply to the orthographic context of a stem and affix at the point of the concatenation or deletion operation. Thus, there are two kinds of spelling rules:

1. Suffix rules, which describe spelling changes applying to the end of the stem and the beginning of the suffix, and

2. Prefix rules, which describe spelling changes applying to the end of the prefix and the beginning of the stem.

A spelling rule can make reference to literal strings and variables. A variable refers to a named set of characters and/or strings, such as *Vowel* (a,e,i,o,u) or *Dental* (d,t,dn,tn,chn,fn,gn). The grammar writer may define such sets and variables ranging over those sets.

The general form of a suffix spelling rule is as follows:

(<parameter>*) [<stem-pattern>] <operator>[<affix-pattern>]
--> [<merged pattern>] {<locs>}

The operator may be either + or - , indicating concatenation and deletion respectively. The <merged-pattern> refers to the form constructed by performing the operation on a stem and affix. The two patterns on the left of the arrow refer to the stem and affix participating in the construction. Each pattern is a list of variables and/or literal strings. Whenever the same variable name appears more than once in the rule, it is assumed to take on the same value throughout.

<parameter> is a Boolean condition on the applicability of the spelling rule. It is necessary for those cases where the application of the rule depends on information about the lexical item which is not included in the orthography. (Like [BEAR88], we choose to represent these conditions as features rather than as diacritics [KOSKENNIEMI84].) An example in English where a parameter is necessary is the case of geminating final consonants. Gemination depends on phonological characteristics which are not predictable from the spelling alone. Only words whose lexical entries contain the specified parameter value will undergo spelling changes sensitive to that parameter.

Specifying orthographic rules independently of the specific affixes to which they apply allows for a more concise declarative representation, as regularities across paradigms and forms can be abstracted out. However, there are cases in which the application of an orthographic rule is constrained to specific paradigms or to specific forms within a paradigm. The optional <locs> qualifier can be used to limit the paradigms and/or specific forms to which the orthographic rule applies.

Prefixation rules are expressed in a similar manner, except that the <operator> precedes the first pattern in the left hand side. Stem changes (in which a stem undergoes a spelling change in the absence of any affixation operation) are handled by the association of an orthographic rule with a form rule of the form <form> : <stem>. The orthographic rule in such a case would contain no affix pattern.

Here we illustrate a hypothetical spelling rule:

["a" Cons Cons] +[Vowel] --> "e" Cons Vowel

This is a suffix rule, since the operator precedes the second left-hand-side pattern. Accordingly, the <stem-pattern> refers to the characters at the end of the stem while the <affix-pattern> refers to the letters at the beginning of the affix. This rule states that, if we are appending an affix which begins with a vowel to a stem which ends in the character "a" followed by two identical consonants, then we construct the resulting form (<merged-pattern>) as follows:

1. Remove the last three characters from the stem, leaving <sub-stem>.
2. Remove the first character from the suffix, leaving <sub-affix>.
3. Construct the string <spell-change> by concatenating the strings and instantiated character variables described by the right-hand-side pattern.
4. Construct the resulting form as the concatenation of the strings <sub-stem>, <spell-change>, and <sub-affix>.

2.4 The Lexicon

We have seen above how one paradigm can be based on another, thereby allowing form construction rules to be "inherited" by paradigms. This inheritance is controlled through the form names themselves. If we have a paradigm B based on paradigm A, then any form rules in A for which there is no rule in B with the same form name are by default assumed to be part of paradigm B.

Although our lexicon is maintained as a secondary storage database with entries represented and indexed differently from the (memory resident) paradigms, it is useful to think of a lexical entry as "inheriting" rules from its paradigm as well. The inflectional behavior of any individual word will depend on both the information inherited from its paradigm and the information stored in the lexicon. Lexicon entries contain the equivalent of a single kind of form construction rule:

<form> : <stem> /{supersede | augment}

The interaction of lexical information with the word's paradigm is as follows:

- If <form> corresponds to a lexical stem rule in the paradigm (i.e., one whose right-hand-side is the special symbol LEX), then this form provides the stem for that rule.
- If <form> corresponds to a surface form in the paradigm or an intermediate form qualified with the qualifier /allow_lexical_override , then the lexical form either *supersedes* or *augments* the construction rule in the paradigm, depending on the value of the stem's /{supersede | augment} qualifier.

The qualifier /allow_lexical_override is necessary to inform the run-time inflectional analyzer when to attempt a lexical lookup of an intermediate form stem. By default, the analyzer looks up any form found directly in the text (surface form) and any forms whose right hand side is LEX. The use of the /allow_lexical_override flag can save disk accesses by limiting lexical lookups of intermediate forms to just those cases in which lexical overrides may actually occur.

Utilizing the /allow_lexical_override qualifier and the default lookup of surface forms, one could write lexical entries in which *all* the rules in a paradigm were overridden by lexical information. In general, this is not a good idea, since it fails to take advantage of the generalizations that paradigms provide, but there are exceptional cases, such as the verb "be", for which there must necessarily be a large number of lexical stems. Allowing lexical overrides in this manner eliminates the need to create an excessive number of highly idiosyncratic paradigms specifically to accomodate irregular verbs in languages like French and German (see section 3).

3 Using Paradigm Inheritance to Capture Linguistic Generalizations

In PDL, word formation is characterized as a sequence of discrete transformational steps. In many cases, paradigms (as well as individual lexical items) will differ with respect to one or more of these intermediate steps, yet share the bulk of the rules that apply to the results of the intermediate operations. Default inheritance, including the inheritance of the partially derived forms, makes it possible to express such facts very succinctly. Figure 1 depicts the hierarchy of paradigms we have developed for the French verbs. The root of the hierarchy (VERB_ROOT) represents the "greatest common denominator" of all the paradigms in the hierarchy. (All of the intermediate form rules in the root paradigm are shown in Figure 1, but many of the surface form rules are omitted because of space limitations. However, all of the form rules, both intermediate and surface, in the other paradigms are listed.) The first sub-paradigm, VERB_ER, represents what are commonly referred to as first conjugation verbs, VERB_IR represents the second conjugation, and VERB_RE_IR, VERB_OIR, and VERB_RE together represent the third conjugation, which includes virtually all of the "irregular" verbs.

[BESCHERELLE90] describes over 70 conjugation types that fall within one of the three basic groups, the third group being subdivided into three sections, one for the irregular verbs ending in -ir, one for the -oir verbs and one for the -re verbs. These sections map directly onto paradigms VERB_RE_IR, VERB_OIR, and VERB_RE, respectively, with the exception of several types (which actually fit VERB_ROOT directly.) Through the use of form rule inheritance, intermediate form rules, lexical override and orthographic rules, we are able to condense the rules for the 78 types into these six paradigms, which capture in a straightforward way most of the linguistic regularities within and among the paradigms.

The useful role played by intermediate form rules in inheritance can be seen by comparing the VERB_ER and VERB_IR paradigms. Both share (inherit) the **imp** intermediate form and the set of six surface forms that describe the imperfect tense (e.g., **imp_1s**). However, they differ in the surface form **prés_1p**, which is overridden in VERB_IR, and in the intermediate form **base**, which is overridden in VERB_ER. The interesting point here is that even though the imperfect indicative forms employ the stem **imp**, a form that is generated from a form that is not shared (**prés_1p**) and which is in turn generated from an unshared form (**base**), both the **imp** stem and the set of imperfect indicative forms may still be shared.

Another example of how overridable intermediate form rules can be used to condense paradigms is provided by the VERB_RE_IR paradigm (which handles all of the irregular verbs ending in -ir that behave more like the -re verbs, e.g., *dormir* and *vêtir*) and its sub-paradigms. This is accomplished by first defining a new intermediate form, **prés_s**, which may be overriden by a lexical entry (or stem change rule). This allows for an irregular stem in the singular forms of the present indicative (e.g., *dormir* --> *dor*, *mouvoir* --> *meu*) while not overriding the **base** form, which is used elsewhere. Secondly, allowing lexical override of the stems used to generate the future and present conditional tense forms (**fut**) and the past simple and imperfect subjunctive forms (**pas**), respectively, allows for irregular stems such as *valoir* --> *vaudr* (**fut**) and *mouvoir* --> *mu* (**pas**).

We have found this combination of intermediate form rules and lexical override useful for defining paradigms for German verbs as well. Because some strong verbs undergo a stem change in the 2nd and 3rd person singular forms of the present tense, an additional intermediate form may be defined to accomodate possible stem

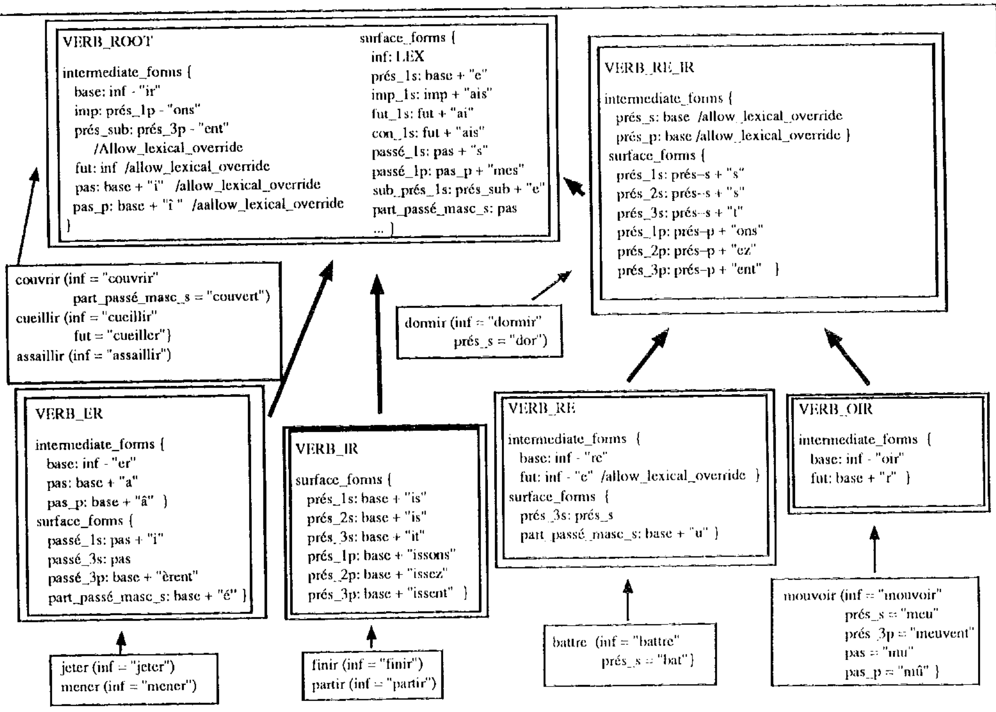

Figure 1. Paradigm inheritance hierarchy for French verbs. Paradigms are surrounded by double boxes. Example lexical items for each paradigm are in single boxes.

changes in these two forms, just as the intermediate form **prés_s** was employed in the French paradigm VERB_RE_IR. This allows all of the strong verbs to be combined into a single paradigm.

4 Compilation and Run-time Algorithms

A PDL description is compiled into a non-deterministic transition network, suitable for the recognition and generation of word forms, as follows. First, the form rules are chained into a network based on the form names appearing in the rules' left and right hand sides. The full set of paradigms to which each form rule applies is calculated and stored at each corresponding node in the network. Then the orthographic rules are conflated with the word formation rules by unifying the orthographic patterns with the affixes in the form rules. Finally, a character discrimination net is constructed from all suffix surface form rules to optimize the run-time matching of the outermost suffix patterns in the form rule transition net.

During morphological analysis, the conflated patterns are matched against the input string and the string undergoes whatever transformation the corresponding word formation rule dictates. At each step through the network, the set of paradigms for which that step is valid is intersected with the set that has been valid up to that point in the derivation. If this intersection becomes NULL, then the path is abandoned as invalid. Traversal through the net proceeds along all possible paths for as long as patterns continue to match. Lexicon lookups of candidate stem strings occur only when a LEX node or node marked as lexically overridable is reached. If a lexical stem matching the form name, paradigm set, and feature constraints acquired from the net is found, then its lemma is returned.

For generation, the traversal is reversed. However, in order to calculate the sequence of rules to traverse to generate a surface form, we must work backwards from the rule that produces the desired surface form (given the paradigm of the lemma) to the rule that precedes that rule, and so on, until we reach a form whose stem is stored with the lemma in the lexicon. At this point, we know both the proper starting lexical stem form and the sequence of rules to apply to that stem.

5 Discussion

A number of researchers have proposed the use of inheritance in representing aspects of natural language (e.g., [HUDSON84], [EVANS89] [DAELEMANS90], [PUSTEJOVSKY91]). The work described here is most similar in spirit to the work of [CALDER89] and [RUSSELL91], who also apply principles of defeasible inheritance to the domain of computational morphology. Calder's word formation rules make use of string equations, an elegant and powerful declarative device which, while more expressive than our (deliberately) constrained word formation and orthographic rules, may be less amenable to efficient compilation and appears geared towards an in-memory lexicon. By disallowing recursion in our form rules, limiting each form rule to at most one affixation operation, and encoding directionality within our orthographic patterns, we are able to compile rules into transition networks in a straightforward manner, reducing the need for extensive run-time unification. In our experience to date, these language limitations have not interfered with the concise capture of morphological behavior. Indeed, our separation of orthographic rules and form rules allows us to capture orthographic generalizations that Calder (1989) cannot. Furthermore, whereas Calder's system "disallows the possibility of inheritance of partial derived string forms," we have found that the inheritance of intermediate stems contributes considerably to the descriptive power of our formalism.

Russell et al ([RUSSELL91]) have developed language extensions to the PATR II style unification grammar formalism which allow for multiple default inheritance in the description of lexical entries. Multiple inheritance is a useful tool for partitioning syntactic, semantic, and morphological classes of behavior. However, while we have encountered occasional cases in which a word appears to derive variants from multiple paradigms, we have so far opted to preserve the simplicity of a single inheritance hierarchy in PDL, utilizing extra lexical stems to accomodate such variants when they arise.

Byrd and Tzoukermann ([BYRD88]) note that their French word grammar contains 165 verb stem rules and another 110 affix rules; and they question the relative value of storing rules versus inflected forms. This is a concern of ours as well, as we wish to minimize the number of run-time "false alarms", potential stems generated during morphological analysis which do not actually exist in the lexicon. Our model of the French verb inflections uses 81 form rules and 17 orthographic rules. We have tried to design our paradigms to minimize the number of inflected stems that must be stored in the lexicon, while at the same time avoiding rules that would contribute to a proliferation of false alarms during analysis. We believe that the use of lexically overridable intermediate forms is a key to striking this balance.

For the purpose of acquiring morphological information about unknown words in a corpus, however, it is useful to have a single canonical form (citation form) for each paradigm, from which all inflected forms in the paradigm can be derived. Thus we have opted to extend our language with the notion of "acquisition-only" paradigms. These paradigms are essentially the same as those used for recognition; however, they include extra form rules (typically stem change rules) to reduce all lexical stems within a paradigm to a single citation stem. The inheritance provisions of PDL make it very easy to add such paradigms. However, any lemma created during the acquisition procedure using an acquisition-only paradigm must be mapped to its equivalent lemma based on the corresponding recognition-time paradigm. This involves generating the extra lexical stems required by the recognition-time paradigm, so that these stems, in addition to the citation stem, can be stored directly in the lexicon.

Several traditionally problematic aspects of German morphology have proved problematic for our formalism as well and we have adopted extensions to the language to accomodate them. Modeling the stem changes involving German "Umlautung" ([TROST90]) has required the addition of a variable mapping function to the specification of orthographic rules. German separable prefixes are handled via the use of an affix variable, which retains the value of the separable prefix for later unification with the separable-prefix feature of potential lexical stems. German compounding remains impossible to capture within our current form rules, as they are constrained to a single <stem> component. While we expect to store most compounds directly in the lexicon, we are looking into heuristics for analyzing compounds that minimize the number of probes needed into our secondary storage lexicon.

6 Conclusions

Our experience so far with PDL has supported our hypothesis that organizing morphological behavior in terms of hierarchically related inflectional paradigms helps to explicitly characterize the similarities and differences among classes of words and makes it easier to capture in a concise and transparent manner the kinds of word formation rules described in many grammar books. The language can be compiled into a form amenable to efficient analysis and generation with a dynamic secondary storage lexicon. Future work includes further "tuning" of existing rulesets, extending our coverage of European languages, and interfacing the inflectional system with rules of derivational morphology and compounding.

Acknowledgements

The authors gratefully acknowledge the invaluable assistance of Alain Couillault, Hengâmeh Irandoust, and Michael Carl in developing grammars for French and German. We also wish to thank Rex Flynn, David Hanssen, and Mayank Prakash for their thoughtful feedback on our design of PDL.

References

[ANICK90] Anick, P. G., J. D. Brennan, R. A. Flynn, D. R. Hanssen, B. Alvey and J. M. Robbins. A Direct Manipulation Interface for Boolean Information Retrieval via Natural Language Query, in Proceedings of ACM/SIGIR '90, Brussels, 1990.

[BEAR88] Bear, John. Morphology with Two-Level Rules and Negative Rule Features, in Proceedings of COLING '88, Budapest, 1988.

[BESCHERELLE90] Bescherelle. La Conjugaison 12000 Verbes. Hatier: Paris, 1990.

[BOBROW77] Bowbrow, D. G. and T. Winograd. An Overview of KRL, a Knowledge Representation Language. Cognitive Science, 1:3-46, 1977.

[BYRD88] Byrd, Roy J. and Evelyne Tzoukermann. Adapting an English Morphological Analyzer for French, in Proceedings of ACL, 1988.

[CALDER89] Calder, Jonathan. Paradigmatic Morphology, in Proceedings of the 4th EACL, Manchester, 1989.

[DAELEMANS90] Daelemans, W. and G. Gazdar (eds.) Inheritance in Natural Language Processing: Workshop Proceedings. ITK, Tilburg University, 1990.

[EVANS89] Evans, Roger and Gerald Gazdar. Inference in DATR, in Proceedings of the 4th EACL, Manchester, 1989.

[GÖRZ88] Görz, Günther and Dietrich Paulus. A Finite State Approach to German Verb Morphology, in Proceedings of COLING '88, Budapest, 1988.

[HUDSON84] Hudson, Richard. Word Grammar. Basil Blackwell: Oxford, 1984.

[KOSKENNIEMI84] Koskenniemi, K. A General Computational Model for Word-form Recognition and Production, in Proceedings of COLING '84, Stanford, 1984.

[NEBEL74] Nebel, Cécile and Frederick F. Fales. French Grammar. Monarch Press: New York, 1974.

[PUSTEJOVSKY91] Pustejovsky, James. The Generative Lexicon. Computational Linguistics, 17(4), 1991.

[RUSSELL91] Russell, Graham, John Carroll, and Susan Warwick-Armstrong. Multiple Default Inheritance in a Unification-Based Lexicon, in Proceedings of ACL, Berkeley, 1991.

[TROST90] Trost, Harald. The Application of Two-level Morphology to Non-concatenative German Morphology, in Proceedings of COLING '90, Helsinki, 1990.