PARSING A FREE-WORD ORDER LANGUAGE: WARLPIRI

Michael B. Kashket
Artificial Intelligence Laboratory
Massachusetts Institute of Technology
545 Technology Square, room 823
Cambridge, MA 02139

ABSTRACT

Free-word order languages have long posed significant problems for standard parsing algorithms. This paper reports on an implemented parser, based on Government-Binding theory (GB) (Chomsky, 1981, 1982), for a particular free-word order language, Warlpiri, an aboriginal language of central Australia. The parser is explicitly designed to transparently mirror the principles of GB.

The operation of this parsing system is quite different in character from that of a rule-based parsing system, *e.g.*, a context-free parsing method. In this system, phrases are constructed via principles of selection, case-marking, case-assignment, and argument-linking, rather than by phrasal rules.

The output of the parser for a sample Warlpiri sentence of four words in length is given. The parser was executed on each of the 23 other permutations of the sentence, and it output equivalent parses, thereby demonstrating its ability to correctly handle the highly scrambled sentences found in Warlpiri.

INTRODUCTION

Basing a parser on Government-Binding theory has led to a design that is quite different from traditional algorithms.[1] The parser presented here operates in two stages, lexical and syntactic. Each stage is carried out by the same parsing engine. The lexical parser projects each constituent lexical item (morpheme) according to information in its associated lexical entries. Lexical parsing is highly data-driven from entries in the lexicon, in keeping with GB. Lexical parses returned by the first stage are then handed over to the second stage, the syntactic parser, as input, where they are further projected and combined to form the final phrase marker.

Before plunging into the parser itself, a sample Warlpiri sentence is presented. Following this, the theory of argument (*i.e.*, NP) identification is given, in order to show how its substantive linguistic principles may be used directly in parsing. Both the lexicon and the other basic data structures are then discussed, followed by a description of the central algorithm, the parsing engine. Lexical phrase-markers produced by the parser for the words *kurduku* and *puntarni* are then given. Finally, the syntactic phrase-marker for the sample sentence is presented. All the phrase-markers shown are slightly edited outputs of the implemented program.

[1] Johnson (1985) reports another design for analyzing discontinuous constituents; it is not grounded on any linguistic theory, however.

A SAMPLE SENTENCE

In order to make the presentation of the parser a little less abstract, a sample sentence of Warlpiri is shown in (1):

(1) Ngajulu-rlu ka-rna-rla punta-rni kurdu-ku karli.
I-ERG PRES-1-3 take-NPST child-DAT boomerang
'I am taking the boomerang from the child.'

(The hyphens are introduced for the nonspeaker of Warlpiri in order to clearly delimit the morphemes.) The second word, *karnarla*, is the auxiliary which must appear in the second (Wackernagel's) position. Except for the auxiliary, the other words may be uttered in any order; there are 4! ways of saying this sentence.

The parser assumes that the input sentence can be broken into its constituent words and morphemes.[2] Sentence (1) would be represented as in (2). The parser can not yet handle the auxiliary, so it has been omitted from the input.

(2)
```
((NGAJULU RLU) (PUNTA RNI) (KURDU KU) (KARLI))
```

ARGUMENT IDENTIFICATION

Before presenting the lexicon, GB argument identification as it is construed for the parser is presented.[3] Case is used to identify syntactic arguments and to link them to their syntactic predicates (*e.g.*, verbal, nominal and infinitival). There are three such cases in Warlpiri: ergative, absolutive and dative.

Argument identification is effected by four subsystems involving case: selection, case-marking, case-assignment, and argument-linking. Only maximal projections (*e.g.*, NP and VP, in English) are eligible to be arguments. In order

[2] Barton (1985) has written a morphological analyzer that breaks down Warlpiri words in their constituent morphemes. We have connected both parsers so that the user is able to enter sentences in a less stilted form. Input (2), however, is given directly to the main parser, bypassing Barton's analyzer.

[3] This analysis of Warlpiri comes from several sources, and from the helpful assistance of Mary Laughren. See, for example, (Laughren, 1978; Nash, 1980; Hale, 1983).

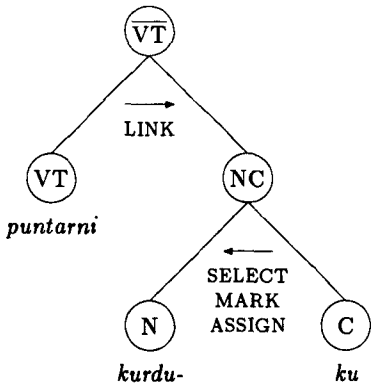

Figure 1: An example of argument identification.

for such a category to be identified as an argument, it must be visible to each of the four subsystems. That is, it must qualify to be selected by a case-marker, marked for its case, assigned its case, and then linked to an argument slot demanding that case.

Selection is a directed action that, for Warlpiri, may take the category preceding it as its object. This follows from the setting of the head parameter of GB: Warlpiri is a head-final language. Selection involves a co-projection of the selector and its object, where both categories are projected one level. For example, the tensed element, *rni*, selects verbs, and then co-projects to form the combined "inflected verb" category. An example is presented below. The other three events occur under the undirected structural relation of siblinghood. That is, the active category (*e.g.*, case-marker) must be a sibling of the passive category (*e.g.*, category being marked for the case).

Consider figure 1. The dative case-marker, *ku*, selects its preceding sibling, *kurdu*, for dative case. Once co-projected, the dative case-marker may then mark its selected sibling for dative case. Because *ku* is also a case-assigner, and because *kurdu* has already been marked for dative case, it may also be assigned dative case. The projected category may then be linked to dative case by *punta-rni* which links dative arguments to the source thematic (θ) role because it has been assigned dative case. In this example, the dative case-marker performed the first three actions of argument identification, and the verb performed the last. Note that only when *kurdu* was selected for case was precedence information used; case-marking, case-assignment and argument-linking are not directional. In this way, the fixed-morpheme order and free-word order have been properly accounted for.

THE LEXICON

The actions for performing argument identification, as well as the data on which they operate, are stored for each lexical item in the lexicon. The part of the lexicon necessary to parse sentence (2) is given in figure 2.

The lexicon is intended to be a transparent encoding

```
(KARLI (datum (v -))
       (datum (n +)))
(KU (action (assign dative))
    (action (mark dative))
    (action
      (select (dative ((v . -) (n . +)))))
    (datum (case dative))
    (datum (percolate t)))
(KURDU (datum (v -))
       (datum (n +)))
(NGAJULU (datum (v -))
         (datum (n +))
         (datum (person 1))
         (datum (number singular)))
(PUNTA (datum (v +))
       (datum (n -))
       (datum (conjugation 2))
       (datum
         (theta-roles (agent theme source))))
(RLU (action (mark ergative))
     (action
       (select (ergative ((v . -) (n . +)))))
     (datum (case ergative))
     (datum (percolate t)))
(RNI (action (assign absolutive))
     (action
       (select (+ ((v . +) (n . -)
                   (conjugation . 2)))))
     (datum (tns +))
     (datum (tense nonpast)))
```

Figure 2: A portion of the lexicon.

of the linguistic knowledge. CONJUGATION stands for the conjugation class of the verb; in Warlpiri there are five conjugation classes. SELECT takes a list of two arguments. The first is the element that will denote selection; in the case of a grammatical case-marker, it is the grammatical case. The second argument is the list of data that the prospective object must match in order to be selected. For example, *rlu* requires that its object be a noun in order to be selected.

The representation for a lexicon is simply a list of morpheme-value pairs; lookup consists simply of searching for the morpheme in the lexicon and returning the value associated with it. The associated value consists of the information that is stored within a category, namely, data and actions. Only the information that is lexically determined, such as person and number for pronouns, is stored in the lexicon.

There is another class of lexical information, lexical rules, which applies across categories. For example, all verbs in Warlpiri with an agent θ-role assign ergative case. Since this case-assignment is a feature of all verbs, it would not be appropriate to store the action in each verbal entry; instead, it stated once as a rule. These rules are represented straightforwardly as a list of pattern-action pairs. After lexical look-up is performed, the list of rules is applied. If the pattern of the rule matches the category, the rule fires, *i.e.*, the information specified in the "action" part of the rule is added to the category. For an example, see the parse of the inflected verb, *puntarni*, in figure 4, below.

THE BASIC DATA STRUCTURES

The basic data structure of the parsing engine is the *projection*, which is represented as a tree of categories. Both dominance and precedence information is recorded explicitly. It should be noted, however, that the precedence relations are not considered in all of the processing; they are taken into account only when they are needed, *i.e.*, when a category is being selected.

While the phrase-marker is being constructed there may be several independent projections that have not yet been connected, as, for example, when two arguments have preceded their predicate. For this reason, the phrase-marker is represented as a forest, specifically with an array of pointers to the roots of the independent projections. An array is used in lieu of a set because the precedence information is needed sometimes, *i.e.*, when selecting a category, as above.

These two structures contain all of the necessary structural relations for parsing. However, in the interests of explicit representation and speeding up the parser somewhat, two auxiliary structures are employed. The *argument set* points to all of the categories in the phrase-marker that may serve as arguments to predicates. Only maximal projections may be entered in this set, in keeping with $\overline{X}$-theory. Note that a maximal projection may serve as an argument of more than one predicate, so that a category is never removed from the argument set.

The second auxiliary structure is the *set of unsatisfied predicates*, which points to all of the categories in the phrase-marker that have unexecuted actions. Unlike the argument set, when the actions of a predicate are executed, the category is removed from the set.

The phrase-marker contains all of the structural relations required by GB; however, there is much more information that must be represented in the output of the parser. This information is stored in the feature-value lists associated with each category. There are two kinds of features: data and actions. There may be any number of data and actions, as dictated by GB; that is, the representation does not constrain the data and actions. The actions of a category are found by performing a look-up in its feature-value list. On the other hand, the data for a category are found by collecting the data for itself and each of the subcategories in its projection in a recursive manner. This is done because data are not percolated up projections.

The list of actions is not completely determined. Selection, case-marking, case-assignment, and argument linking are represented as actions (*cf.* the discussion of case, above). It should be noted that these are the only actions available to the lexicon writer. Actions do not consist of arbitrary code that may be executed, such as when an arc is traversed in an ATN system. The supplied actions, as derived from GB, should provide a comprehensive set of linguistically relevant operations needed to parse any sentence of the target language.

Although the list of data types is not yet complete, a few have already proved necessary, such as person and number information for nominal categories. The list of θ-roles for which a predicate subcategorizes is also stored as data for the category.

THE PARSING ENGINE

The parsing engine is the core of both the lexical and the syntactic parsers. Therefore, their operations can be described at the same time. The syntactic parser is just the parsing engine that accepts sentences (*i.e.*, lists of words) as input, and returns syntactic phrase-markers as output. The lexical parser is just the parsing engine that accepts words (*i.e.*, lists of morphemes) as input, and returns lexical phrase-markers as output.

The engine loops through each component of the input, performing two computations. First it calls its subordinate parser (*e.g.*, the lexical parser is the subordinate parser of the syntactic parser) to parse the component, yielding a phrase-marker. (The subordinate parser for the lexical parser performs a look-up of the morpheme in the lexicon.) In the second computation, the set of unsatisfied predicates is traversed to see if any of the predicates' actions can

apply. This is where selection, case-marking, projection, and so on, are performed.

Note that there is no possible ambiguity during the identification of arguments with their predicates. This stems from the fact that selection may only apply to the (single) category preceding the predicate category, and that each of the subsequent actions may only apply serially. This assumes single-noun noun phrases. In the next version of the parser, multiple-noun noun phrases will be tackled. However, the addition of word stress information will serve to disambiguate noun grouping.

There may be ambiguity in the parsing of the morphemes. That is, there may be more than one entry for a single morpheme. The details of this disambiguation are not clear. One possible solution is to split the parsing process into one process for each entry, and to let each daughter process continue on its own. This solution, however, is rather brute-force and does not take advantage of the limited ambiguity of multiple lexical entries. For the moment, the parser will assume that only unambiguous morphemes are given to it.

After the loop is complete, the engine performs default actions. One example is the selection for and marking of absolutive case. In Warlpiri, the absolutive case-marker is not phonologically overt. The absolutive case-marker is left as a default, where, if a noun has not been marked for a case upon completion of lexical parsing, absolutive case is marked. This is how *karli* is parsed in sentence (2); see figures 6 and 7, below.

The next operation of the engine is to check the well-formedness of the parse. For both the lexical parser and the syntactic parser, one condition is that the phrase-marker consist of a single tree, *i.e.*, that all constituents have been linked into a single structure. This condition subsumes the Case Filter of GB. In order for a noun phrase to be linked to its predicate it must have received case; any noun phrase that has not received case will not be linked to the projection of the predicate, and the phrase-marker will not consist of a single tree.

The last operation percolates unexecuted actions to the root of the phrase-marker, for use at the next higher level of parsing. For example, the assignment of both ergative case and absolutive case in the verb *puntarni* are not executed at the lexical level of parsing. So, the actions are percolated to the root of the phrase-marker for the conjugated verb, and are available for syntactic parsing. In the parse of sentence (2), they are, in fact, executed at the syntactic level.

TWO PARSED WORDS

The parse of *kurduku*, meaning 'child' marked for dative case, is presented in figure 3. It consists of a phrase-marker with a single root, corresponding to the declined noun. It has two children, one of which is the noun, *kurdu*, and the other the case-marker, *ku*.

```
0: actions: ASSIGN: DATIVE
            MARK: DATIVE
            SELECT: (DATIVE ((V . -) (N . +)))
   projection?: NIL
   children: 0: data: ASSIGN: DATIVE
                      MARK: DATIVE
                      SELECT: DATIVE
                      TIME: 1
                      MORPHEME: KURDU
                      N: +
                      V: -
                 projection?: T
             1: data: TIME: 2
                      MORPHEME: KU
                      PERCOLATE: T
                      CASE: DATIVE
                 projection?: T
```

Figure 3: The parse of *kurduku*.

One can see that all three actions of the case-marker have executed. The selection caused the noun, *kurdu*, and the case-marker, *ku*, to co-project; furthermore, the noun was marked as selected (SELECT: DATIVE appears in its data). Marking and assignment also are evident. Note that all three actions percolated up the projection. This is due to the PERCOLATE: T datum for *ku*, which forces the actions to percolate instead of simply being deleted upon execution. The actions of case-markers percolate because they can be used in complex noun phrase formation, marking nouns that precede them at the syntactic level. This phenomenon has not yet been fully implemented. The TIME datum is used simply to record the order in which the morphemes appeared in the input so that the precedence information may be retained in the parse. One more note: the PROJECTION? field is true when the category's parent is a member of its projection, and false when it isn't. Because the top-level category in the phrase-marker is a projection of both subordinate categories, the PROJECTION? entries for both of them are true.

In figure 4, the parse of *puntarni* is shown. There is much more information here than was present for each of the lexical entries for the verb, *punta*, and the tensed element, *rni*. The added information comes from the application of lexical rules, mentioned above. These rules first associate the θ-roles with their corresponding cases, as can be seen in the data entry for *punta*. Second, they set up the INTERNAL and EXTERNAL actions which project one and two levels, respectively, in syntax. That is, the agent, which will be marked with ergative case, will fill the subject position; the theme and the source, which will be marked with absolutive and dative cases, will fill the object positions.

```
0: actions: ASSIGN: ABSOLUTIVE
            INTERNAL: SOURCE
            INTERNAL: THEME
            EXTERNAL: AGENT
            ASSIGN: ERGATIVE
   projection?: NIL
   children: 0: data: SELECT: +
                      TIME: 1
                      THEME: ABSOLUTIVE
                      SOURCE: DATIVE
                      AGENT: ERGATIVE
                      MORPHEME: PUNTA
                      THETA-ROLES:
                        (AGENT THEME SOURCE)
                      CONJUGATION: 2
                      N: -
                      V: +
                 projection?: T
             1: data: TIME: 2
                      MORPHEME: RNI
                      TENSE: NONPAST
                      TNS: +
                 projection?: T
```

Figure 4: The parse of *puntarni*.

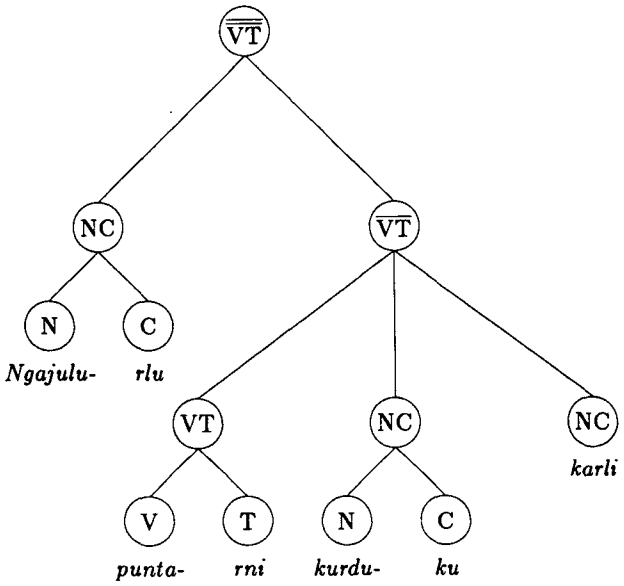

Figure 5: The phrase-marker for sentence (2).

A PARSED SENTENCE

The phrase-marker for sentence (2) is given in figure 5. The corresponding parse for this sentence is shown in figures 6 and 7, the actual output of the parser. In the parse, the verb has projected two levels, as per its projection actions, INTERNAL and EXTERNAL. These two actions are particular to the syntactic parser, which is why they were not executed at the lexical level when they were introduced. INTERNAL causes the verb to project one level, and inserts the LINK action for the object cases. EXTERNAL causes a second level of projection, and inserts the LINK action for the subject case. Note that the TIME information is now stored at the level of lexical projections; these are the times when the lexical projections were presented to the syntactic parser.

To demonstrate the parser's ability to correctly parse free word order sentences, the other 23 permutations of sentence (2) were given to the parser. The phrase-markers constructed, omitted here for the sake of brevity, were equivalent to the phrase-marker above. That is, except for the ordering of the constituents, the domination relations were the same: the noun marked for ergative case was in all cases the subject, associated with the agent θ-role; and the nouns marked for absolutive and dative cases were in all cases the objects, associated with the theme and source θ-roles, respectively.

CONCLUSION

We have presented a currently implemented parser that can parse some free-word order sentences of Warlpiri. The representations (*e.g.*, the lexicon and phrase-markers) and algorithms (*e.g.*, projection, undirected case-marking, and the directed selection) employed are faithful to the linguistic theory on which they are based. This system, while quite unlike a rule-based parser, seems to have the potential to correctly analyze a substantial range of linguistic phenomena. Because the parser is based on linguistic principles it should be more flexible and extendible than rule-based systems. Furthermore, such a parser may be changed more easily when there are changes in the linguistic theory on which it is based. These properties give the class of principle-based parsers greater promise to ultimately parse full-fledged natural language input.

```
0: projection?: NIL
   children:
     0: actions: MARK: ERGATIVE
                 SELECT:
                   (ERGATIVE ((V . -) (N . +)))
        data: LINK: ERGATIVE
              ASSIGN: ERGATIVE
              TIME: 1
        projection?: NIL
        children:
          0: data: MARK: ERGATIVE
                   SELECT: ERGATIVE
                   MORPHEME: NGAJULU
                   NUMBER: SINGULAR
                   PERSON: 1
                   N: +
                   V: -
             projection?: T
          1: data: MORPHEME: RLU
                   PERCOLATE: T
                   CASE: ERGATIVE
             projection?: T
     1: projection?: T
        children:
          0: data: TIME: 2
             projection?: T
             children:
               0: data: SELECT: +
                        THEME: ABSOLUTIVE
                        SOURCE: DATIVE
                        AGENT: ERGATIVE
                        MORPHEME: PUNTA
                        THETA-ROLES:
                          (AGENT THEME SOURCE)
                        CONJUGATION: 2
                        N: -
                        V: +
                  projection?: T
               1: data: MORPHEME: RNI
                        TENSE: NONPAST
                        TNS: +
                  projection?: T
```

Figure 6: The first half of the parse of sentence (2).

```
1: actions: ASSIGN: DATIVE
            MARK: DATIVE
            SELECT:
              (DATIVE ((V . -) (N . +)))
   data: LINK: DATIVE
         TIME: 3
   projection?: NIL
   children:
     0: data: ASSIGN: DATIVE
              MARK: DATIVE
              SELECT: DATIVE
              MORPHEME: KURDU
              N: +
              V: -
        projection?: T
     1: data: MORPHEME: KU
              PERCOLATE: T
              CASE: DATIVE
        projection?: T
2: data: LINK: ABSOLUTIVE
         ASSIGN: ABSOLUTIVE
         TIME: 4
         MARK: ABSOLUTIVE
         SELECT: ABSOLUTIVE
         MORPHEME: KARLI
         N: +
         V: -
   projection?: NIL
```

Figure 7: The second half of the parse of sentence (2).

ACKNOWLEDGMENTS

This report describes research done at the Artificial Intelligence Laboratory of the Massachusetts Institute of Technology. Support for the Laboratory's artificial intelligence research has been provided in part by the Advanced Research Projects Agency of the Department of Defense under Office of Naval Research contract N00014-80-C-0505. I wish to thank my thesis advisor, Robert Berwick, for his helpful advice and criticisms. I also wish to thank Mary Laughren for her instruction on Warlpiri without which I would not have been able to create this parser.

REFERENCES

Barton, G. Edward (1985). "The Computational Complexity of Two-level Morphology," A.I. Memo 856, Cambridge, MA: Massachusetts Institute of Technology.

Chomsky, Noam (1981). *Lectures on Government and Binding, the Pisa Lectures*, Dordrecht, Holland: Foris Publications.

Chomsky, Noam (1982). *Some Concepts and Consequences of the Theory of Government and Binding*, Cambridge, MA: MIT Press.

Hale, Ken (1983). "Warlpiri and the Grammar of Non-configurational Languages," *Natural Language and Linguistic Theory*, pp. 5–47.

Johnson, Mark (1985). "Parsing with Discontinuous Constituents," *23rd Annual Proceedings of the Association for Computational Linguistics*, pp. 127–32.

Laughren, Mary (1978). "Directional Terminology in Warlpiri, a Central Australian Language," *Working Papers in Language and Linguistics*, Volume 8, pp. 1–16.

Nash, David (1980). "Topics in Warlpiri Grammar," Ph.D. Thesis, M.I.T. Department of Linguistics and Philosophy.