ON THE APPLICABILITY OF TWO LEVEL MORPHOLOGY TO THE INFLECTION OF HEBREW VERBS

by

A. Lavie, A. Itai, U. Ornan and M. Rimon*

Technical Report #513
July 1988

* IBM Scientific Center, Haifa.
ON THE APPLICABILITY OF TWO LEVEL MORPHOLOGY TO THE INFLECTION OF HEBREW VERBS

Alon Lavie
Alon Itai
Uzzi Ornan
Dept. of Computer Science, Technion.

Mori Rimon
IBM Scientific Center, Haifa.

ABSTRACT

Hebrew, as other Semitic languages, has a rich morphology, observable in part by the complexity of verb inflections. The primary base of verbs in Hebrew is the past third singular form of the verb. From this base, some twenty eight different inflected forms can be created according to tense, person, gender and number.

Traditionally, inflection tables were used to describe the various inflected forms derived from the verb base. Research done by Ornan has managed to describe the verb inflection process using the principles of Generative Grammar. In this approach, inflected verb forms are viewed as constructs of the form prefix+base+suffix. Verb inflection is described as a series of sequential operations. The first stage converts the primary verb base to a secondary base, when the secondary base is not the same as the primary base. Secondly, the appropriate prefix and/or suffix are concatenated to the base. Thirdly, several morpho-phonemic changes due to the affix concatenation occur. Algorithms for both generation and analysis according to these principles have been developed.

The formalism developed by Koskenniemi, known as Two Level Morphology, proposes describing the morphology of any natural language by a set of lexicons and a set of Two Level rules, which act in parallel, and transform directly between the lexical and surface representations of a word. This alternative formalism has been used to describe the morphology of several languages, including the major European languages. In our work, we examine the applicability of Two Level Morphology to the description of Hebrew verb inflection.

We consider how to apply the Two level framework to the stages of the inflection process of the previous approach. We show that the Two Level model can adequately handle the concatenation of the affixes to the verb base, and the resulting morpho-phonemic changes that occur. However, the first stage of inflection, namely the conversion of the primary base of the verb to the secondary base, is difficult to describe in a natural way using Two Level rules. This requires that the various verb bases be kept in the lexicon system. We propose a modification to the model that can solve this problem.

In the course of our work, we created a new implementation of the Two Level model written in PROLOG. In our implementation, Two Level rules are converted directly into PROLOG predicates, making the use of the finite state automata of the original model unnecessary. We also propose a structural change in the Two Level lexicon system. An outline of our implementation is presented.
1. INTRODUCTION

Morphology, although somewhat ignored for a long time, plays an important role in natural language processing. In the last few years, the importance of morphological analysis and generation as a separate stage in Natural Language systems, has been generally recognized. Hebrew, as other Semitic languages, has a rich morphology, observable in part by the complexity of verb inflections. It is therefore obvious that any computer system for the processing of Hebrew, must include a non-trivial morphological analysis and generation component.

Section two presents a short description of Hebrew verb morphology, and overviews previous methods for dealing with it.

An alternative formalism for dealing with the morphology of any natural language was proposed by Koskenniemi [KOS1], and is known as Two Level Morphology. Koskenniemi's formalism and implementation has been used to describe the morphology of several languages, including the major European ones [KOS1,KART1]. In our work, we examine the applicability of Two Level Morphology to Hebrew, specifically to the inflection of Hebrew verbs. Section three, therefore, gives a short overview of Koskenniemi's formalism and implementation program.

Section four discusses the description of Hebrew verb inflections using Two Level Morphology, and compares it with the major alternative approach that follows the principles of Generative Grammar.

In Koskenniemi's original model, Two Level rules are converted into finite state automata. The Two Level program simulates the operation of these automata in both the generation and analysis procedures. A compiler for converting Two Level rules to automata has been developed by Karttunen, Koskenniemi and Kaplan [KKK1]. In the course of our work, we have developed an implementation of Koskenniemi's model in PROLOG, in which Two Level rules are converted directly to PROLOG predicates, making the automata used in the original implementation unnecessary. A brief description of our implementation is presented in section five.

In section six we propose two major modifications to the Two Level model. The first is the addition of a conversion procedure between lexicon entries and segments of the lexical string. Such a procedure is essential for conversion of primary verb bases to secondary bases in the processing of Hebrew verb inflections. The other modification we propose is in the structure of the Two Level lexicon system. A more thorough description of our Two Level PROLOG program, including an analytic comparison with Koskenniemi's original implementation, will be presented elsewhere.

Section seven summarizes the work done so far, what is yet to be done, and presents our conclusions.
2. HEBREW VERB MORPHOLOGY

In the Hebrew script, vowels are written mostly as signs above and under the regular alphabet letters. In modern Hebrew, as well as several older scripts, these signs are omitted, leaving the task of recognizing the word and its pronunciation to the skilled reader. This forms a major problem for computer recognition and analysis of words written in Hebrew script. As a result, computer processing of Hebrew, written in vowel omitted script, becomes an extremely difficult task. The ambiguity in word recognition makes the analysis process very complicated, and not solvable without the use of semantic means. As a result most morphological processing in vowel omitted Hebrew has concentrated on generation, which is significantly easier. In our work, we use the Phonemic Script, a script that uses the Latin alphabet, which was developed a few years ago by Oman [ORN2]. In this script the vowels are written as their Latin equivalents, and words are generally written according to their theoretic (phonemic) form, disregarding most phonetical changes. The Phonemic Script enables the word to be uniquely recognized according to its written form by any computer program. A program for converting Phonemic Script to normal Hebrew script has been written by Oman.

The base form of verbs in Hebrew (the form in which the verb appears in most dictionaries and lexicons) is the past third singular form. From this base form the inflection according to tense, person, gender and number produces twenty eight different inflected forms of the verb. Figure 1 shows, as an example, all inflected forms of the verb *katab* (the Hebrew verb for *write*).

Traditionally, inflection tables, such as that in figure 1, were used to describe the various inflected forms of a verb. The verbs in Hebrew are normally categorized according to their pattern, described as a sequence of consonants and vowels, with slots scattered among them to be filled by the consonants of the root (a root is an ordered sequence of consonants). There are seven regular patterns of verbs in Hebrew, known by the name of *Binyanim*. For example, verbs that belong to the *Qal* (*Pa&al*) base pattern have the form "CaCaC", where the root consonants take the places of the "C" place holders, such as in *katab*. For each verb pattern (Binyan), a table of all its derived forms, according to tense, person, gender and number can be listed. If all verb bases were inflected regularly, the inflection process could be adequately described by the set of inflection tables for the various base patterns. Generating an inflected form would involve recognizing the verb pattern, and selecting the inflected form from the appropriate inflection table. Analysis could be done similarly by recognizing the inflected form as an inflection of one of the patterns. Unfortunately, there are quite numerous categories of verbs, whose base is irregular in some respect, having an effect on the various forms of inflection. Creating tables for all inflected forms of all possible bases would take an large amount of space, and is quite unnecessary as we shall see.

Research work done by Oman [ORN1,ORN3] has managed to describe the process of Hebrew verb inflection using the principles of Generative Grammar. The process of verb inflection is described as a sequence of rules operating in series, one on the output of its predecessor, beginning with the primary past form of the verb, and resulting in the derived inflected form. The main idea in the generative grammar approach is to view the inflected forms of the verb as constructs of the form *prefix + base + suffix*. For example, the inflected form *tikbu* (you will write) originates from the construct *t + k tob + u*. 
The verb base in the center of this construct may in some cases be different from the primary verb base. Some of the verb patterns (Binyanim) have secondary verb base patterns for the different tenses. The Qal pattern ("CaCaC") has in fact five separate patterns in the different tenses. The Pual pattern ("CuCaCaC") on the other hand, has only one base pattern (the primary base pattern) in all tenses. Figure 2 is a table of the verb bases of the seven regular patterns in the various tenses [EJ1].

The process of verb inflection can be described as the following sequence of operations. The first stage is the conversion of the base from its primary (past third singular) form to a secondary base according to the tense, when the primary base belongs to a pattern, in which the secondary base is not the same as the primary base. Next, the appropriate prefix and/or suffix are determined according to the desired tense, person, gender and number. In some cases one or both of these affixes may be empty. The resulting form is now a construct of the type prefix + base + suffix. Thirdly, a series of morpho-phonemic changes occur, due to the concatenation of the prefix and suffix to the base.

Algorithms for both generation and analysis, based on the principles of inflection described above, have been developed. A problem with generative grammars is that they are not easily reversible. The grammar rules that specify the production process cannot be simply reversed for the purpose of analysis.

Figure 1: Inflected Forms of the verb katab

<table>
<thead>
<tr>
<th>Person</th>
<th>Number</th>
<th>Past Tense</th>
<th>Present Tense</th>
<th>Imperative Tense</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Gender M</td>
<td>Gender F</td>
<td>Gender M</td>
</tr>
<tr>
<td>P1</td>
<td>Singular</td>
<td>katabti</td>
<td></td>
<td></td>
</tr>
<tr>
<td>P2</td>
<td>Singular</td>
<td>katabta</td>
<td>katabt</td>
<td>koteb</td>
</tr>
<tr>
<td>P3</td>
<td>Singular</td>
<td>katab</td>
<td>katba</td>
<td>P2 Singular</td>
</tr>
<tr>
<td>P1</td>
<td>Plural</td>
<td></td>
<td>katabnu</td>
<td>P1 Plural</td>
</tr>
<tr>
<td>P2</td>
<td>Plural</td>
<td>ktabtem</td>
<td>ktabten</td>
<td>P2 Plural</td>
</tr>
<tr>
<td>P3</td>
<td>Plural</td>
<td>katbu</td>
<td></td>
<td>P3 Plural</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Figure 2: Verb Bases of Qal and Pual Patterns

<table>
<thead>
<tr>
<th>Person</th>
<th>Number</th>
<th>Future Tense</th>
<th>Present Tense</th>
<th>Imperative Tense</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Gender M</td>
<td>Gender F</td>
<td>Gender M</td>
</tr>
<tr>
<td>P1</td>
<td>Singular</td>
<td></td>
<td></td>
<td>P2 Singular</td>
</tr>
<tr>
<td>P2</td>
<td>Singular</td>
<td>tiktob</td>
<td>tiktbi</td>
<td>P2 Plural</td>
</tr>
<tr>
<td>P3</td>
<td>Singular</td>
<td>yiktob</td>
<td>tiktob</td>
<td></td>
</tr>
<tr>
<td>P1</td>
<td>Plural</td>
<td></td>
<td>niktob</td>
<td></td>
</tr>
<tr>
<td>P2</td>
<td>Plural</td>
<td>tiktbu</td>
<td>tiktobna</td>
<td></td>
</tr>
<tr>
<td>P3</td>
<td>Plural</td>
<td>yiktbu</td>
<td>tiktobna</td>
<td></td>
</tr>
<tr>
<td>Pattern Name</td>
<td>Primary Base Form</td>
<td>Secondary Base Form</td>
<td></td>
<td></td>
</tr>
<tr>
<td>--------------</td>
<td>-------------------</td>
<td>---------------------</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Qal</td>
<td>CaCaC</td>
<td>CoCeC</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>CCaC</td>
<td></td>
<td></td>
</tr>
<tr>
<td>nip&amp;al</td>
<td>niCCaC</td>
<td>hiC,C,i aCeC</td>
<td></td>
<td></td>
</tr>
<tr>
<td>pi&amp;&amp;el</td>
<td>CiCCeC</td>
<td>CaCCeC</td>
<td></td>
<td></td>
</tr>
<tr>
<td>pu&amp;&amp;al</td>
<td>CuCCaC</td>
<td>none</td>
<td></td>
<td></td>
</tr>
<tr>
<td>hitpa&amp;&amp;el</td>
<td>hiCCiC</td>
<td>hitCaCCeC</td>
<td></td>
<td></td>
</tr>
<tr>
<td>hip&amp;il</td>
<td>hiCGiC</td>
<td>haCCiC</td>
<td></td>
<td></td>
</tr>
<tr>
<td>hup&amp;al</td>
<td>huCCaC</td>
<td>none</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Remarks:
1. C is a place holder for a root consonant.
2. In the patterns pi\&\&el, pu\&\&al and hitpa\&\&el, when the root consonant slots are filled with a three consonant root, the second consonant is geminated.

Figure 2: Primary and Secondary Bases of Hebrew Verbs.

However, the algorithm developed for the analysis of Hebrew verbs is quite similar to the generation algorithm in the reverse order. The prefixes and suffixes can quite easily be identified, and the original verb base can then be restored. Analysis is nevertheless more complex than generation, and the possibility of more than one correct analysis must be considered.

3. THE TWO LEVEL MODEL

A conceptually new model for the description of natural language morphology was proposed by Koskenniemi in 1983. In his Ph.D. dissertation [KOSI], Koskenniemi presented a new formalism for the description of the morphological changes that occur in words, as well as a computer program implementing the proposed model. His model is independent of the specific language, the morphology of which is to be described, and is supposedly fit to deal with the morphology of any natural language.

The Two Level model, as it has been known, proposes to describe the morphology of a language as a relation between two levels of representation: the lexical level and the surface level. The model is based on two major components. The first is a system of linked lexicons. Entries from these lexicons are concatenated one to another, constructing the lexical representation of a morphological word description. Lexicons exist for various prefixes and suffixes of words, and the word base stems themselves may be divided into several lexicons. The second major component of the model is a set of morphological rules called "Two Level" rules. These rules describe the relationship between characters of the lexical level and characters of the surface level, the level in which the written word is obtained. This binding of lexical characters to
surface characters is on a one to one basis, and is sensitive to the context of both previous and
subsequent pairs of characters appearing in the Two Level strings.

The following example is taken from Karttunen and Wittenburg’s Two Level morphologi­
cal analysis of English [KART2]. Let us consider the following morphological rule in English,
concerning the replacement of "i" with "y", such as in the word *lying*. The lexical representation
of the word has the form *lie + ing*, that is; the verb base *lie* appended by the suffix *ing*. The surface string in this case has the form *by00ing*. Figure 3 shows the appropriate two level strings for this example.

<table>
<thead>
<tr>
<th>Lexical Level</th>
<th>Surface level</th>
</tr>
</thead>
<tbody>
<tr>
<td>lie</td>
<td>ly00ing</td>
</tr>
<tr>
<td>+</td>
<td>0</td>
</tr>
<tr>
<td>ing</td>
<td>0</td>
</tr>
<tr>
<td>+</td>
<td>i</td>
</tr>
<tr>
<td></td>
<td>y</td>
</tr>
<tr>
<td></td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>ing</td>
</tr>
</tbody>
</table>

Figure 3: Two Level string representation.

The "i replacement" rule specifies that the character "i" on the lexical level corresponds to
the surface character "y" in the context that it is followed by the sequence of pairs "[e,0],[+,0],[i,i]". This in fact means that lexical "i" is to be replaced by surface "y" if and only if it is followed by a lexical "e" replaced by the surface null character, followed by a "[+,0]" pair and a lexical “i” that remains on the surface level. In all other cases lexical "i" remains as "i" on
the surface level. Therefore lexical *lie + ing* is realized as *by00ing* at the surface level, becoming *lying* after surface null characters are omitted, but *lie + ed* is realized as *li00ed*, finally becoming *lied* after null characters are deleted.

In his work Koskenniemi proposed a formal language for describing Two Level morpholog­
ic rules. In principle, the idea of these rules is that they all act simultaneously and in parallel,
and that together they describe the relationship that holds between lexical and surface strings. If
we look at both lexical and surface levels simultaneously, we can view the rules as validating
those pairs of strings, in which the lexical string fits the surface string, and rejecting all other
pairs of strings. However, when we are concerned with generation and analysis, only one of the
two levels is in hand. The Two Level program uses the rules to construct the other level. The created string must be such, that combined together, both strings satisfy all Two Level rules.

In his model, Koskenniemi proposes converting the Two Level rules into a special kind of
finite state automata, whose input is a string of pairs of characters instead of single characters.
For each Two Level rule, an automaton is built, so that a pair of strings is accepted by the auto­
maton if and only if the pair of strings is validated by the corresponding Two Level rule. A pair
of strings is accepted if accepted by each and every automaton. The Two Level morphology
implementation program, written by Koskenniemi as part of the model, uses the automata in both
the generation and analysis processes in order to construct the string of one level from the other.
Koskenniemi’s original implementation program contains no compiler for converting Two Level
rules in the proposed formalism into automata. In his work, Koskenniemi sketches how this task
can be done by hand. Such a compiler was later developed by Karttunen, Koskenniemi and
Kaplan [KKK1], but is not yet generally available.
4. A TWO LEVEL DESCRIPTION OF HEBREW VERB INFLECTIONS

4.1 THE APPLICABILITY QUESTION

When we consider the applicability of Two Level morphology to Hebrew, we must take into consideration the fact that inflected verb forms in Hebrew are not completely concatenative in nature. This is a property not only of Hebrew, but of other Semitic languages as well, as stated by Kay [KAY1]. In his work, Kay considers how to apply finite state morphology to Arabic. There exists a great resemblance between the inflection of verbs in Hebrew and Arabic. Kay proposes how a finite state system composed of four separate tiers can handle the inflection of verbs in Arabic. Two tiers are used to represent the verb pattern (a separate tier for pattern consonants and pattern vowels), a third tier for the root consonants, and a forth tier for the inflected form. Kay's system starts, therefore, from a morphological level in which the root and the pattern are represented separately. It is not clear why the pattern representation requires two tiers. In any case, this level of representation, namely the root and pattern represented separately, is deeper than the level we need in our work. The semantic meaning of a verb is determined on the level of the primary base of the verb, which is the combination of both root and pattern. For our purposes, a two tier representation, such as in the Two Level model, should therefore be sufficient.

An inspection of the Hebrew verb inflection process, as described by the generative grammar approach, yields some interesting results. Secondary verb bases, derived from the primary verb base in the first stage of inflection, are in some cases quite different from the primary base. However, the other changes that occur in the inflection process are concatenative in nature. They involve the concatenation of various prefixes and/or suffixes to the verb base and some consequent morpho-phonemic changes in the verb form. We will, therefore, try to apply the concepts of Two Level morphology to the stages of verb inflection separately, and then try to combine them. In order to describe the process of Hebrew verb inflection using the concepts of the Two Level model, we must first determine our intended representations for the lexical and surface levels. It is quite clear that on the lexical level we would like to obtain constructs of the form prefix + base + suffix, where each part of the construct is taken from an appropriate lexicon. On the surface level, we wish to see the final inflected form, possibly with some null characters still to be deleted.

4.2 PRIMARY TO SECONDARY BASE TRANSFORMATION

When we compare the secondary bases to their original primary bases we find that in some cases the two differ in a way, that cannot always be accounted for by local morphological changes. For example, the primary base of the pa\&al (Qal) verb pattern is of the form "CaCaC", where the root consonants fill in the C place holders, such as in katab (he wrote). The present base of this pattern is of the form "CoCeC", where the three C place holders are, again, filled by the root consonants, such as in koteb (you/he is writing). Note that inflecting katab into koteb requires no affix concatenation (both prefix and suffix are empty in this case). It would be difficult to describe the transformation between the primary base and the secondary base in such a case, using Two Level rules that specify the relationship on a character to character basis. Since the change is due to a grammatic cause, the tense, and not to a local morpho-phonemic environment, we would have to add a new character on the lexical level, to trigger the appropriate rules
of transformation. Furthermore, the Two Level model requires a direct transformation between lexical and surface strings. Therefore, we would have to combine the rules that convert the primary base to the secondary one, with the rules that describe the other morphological changes in the word. However, these morphological changes greatly depend on the pattern of the secondary base. It is therefore essential that the secondary base and not the primary base be a component on the lexical level.

It is possible to describe the conversion between the primary and secondary bases using Two Level rules. However, the Two Level rules are not the natural way to describe this first stage of the verb inflection process. The only alternative choice that the Two Level formalism offers, is to keep all bases (primary and secondary) in the lexicon system. This is in fact what Koskenniemi himself does in the analysis of Finnish morphology, described in his work [KOSKI]. For each verb in Finnish, there exist up to four different stems in the stem lexicon, each fitting different suffixes to the verb. However, in Hebrew, the transformation between the primary base and the secondary base is regular in the sense, that it is almost uniquely defined by the pattern of the primary base (the "Binyan") and the tense. Therefore, since there exists such an obvious similarity between the bases of verbs of the same pattern, it seems wasteful to save all the secondary bases of each verb in the lexicon. We discuss a possible supplement to the Two Level model, that can adequately solve this problem, along with some other disadvantages of the currently defined lexicon system, in section six.

4.3 AFFIX ORIENTED MORPHOLOGICAL CHANGES

In the second stage of the inflection process, prefixes and suffixes that depend on tense, person, number and gender are attached to the verb base. This concatenation of affixes to the base results in some morphological changes in the word that are local in nature and can be adequately described by Two Level rules. We shall now present two such examples.

Example 1: The first example is a morphological rule that deals with the place of stress in the verb. The normal stress, in most Hebrew words, is on the last vowel of the word. Some of the suffixes that are added to the verb in the course of inflection, are in fact stressed suffixes. In most cases, when a stressed suffix that starts with a vowel is added to a verb base, the stress that was previously on the last vowel in the verb base disappears, and the last vowel in the verb base is deleted. For example, koteb + im (we/you/they are writing) results in kotbim. Since im is a stressed suffix that starts with the vowel i, the last vowel in the base koteb, namely e, is deleted. Using Koskenniemi's formalism for Two Level rules we can describe this rule in the following way. We use the non-terminal character E to denote an e vowel on the lexical level that may be changed or deleted on the surface level. The rule then takes the following form:

\[ [E,0] \leftrightarrow [cset] \_ [cset, [+1, 0],[',0],vset] \]

The terms *cset* and *vset* denote character pairs, in which the lexical level character is the same as the surface character, and the character belongs to the set of consonants or the set of vowels accordingly. The rule states in fact, that lexical "E" is realized as surface null if and only if it is preceded by a consonant and followed by the sequence of a consonant and a suffix that starts with
a stressed vowel. The "'" character indicates the stressing of the suffix. As a result the lexical form kotEb (I/you/he is writing) will match the surface form koteb (no E deletion) while the lexical string kotEb+'im (we/you/they are writing) will match the surface string kotob00im, which becomes kotbim after null characters are deleted (lexical E is deleted due to the suffix).

Example 2: The second example is of a rule that prevents the formation of a three consonant sequence. The future secondary base of the Qal verb pattern is of the form "CCoC" such as in ktab. When a future consonant prefix such as "t" is attached to the base, a three consonant sequence occurs. Hebrew phonology does not allow such a sequence, and therefore, a vowel must be inserted to break the sequence. The vowel "i" is inserted after the first consonant in the three consonant sequence at the beginning of the word. A Two Level rule that describes this has the form:

\[ [+i] \leftrightarrow [fset] \_ [cset,cset] \]

The rule states that lexical "+i" is realized as the vowel "i" if and only if it is preceded by a future prefix such as "t" and followed by the sequence of two consonants. Therefore the lexical string t+ktab (you will write) matches the surface string tiktob.

We have developed a small set of Two Level rules that describe the morphological changes that occur in verbs when inflected in the present tense. Most of these rules, such as the stress shifting rule, apply to verb inflections in the other tenses as well, and even to noun inflections. We are now working on extending the Two Level rules to deal with verb inflections in the other tenses. We do not expect that the set of rules will grow substantially.

4.4 THE HEBREW LEXICON SYSTEM

The system of lexicons, that we have constructed for the Two level handling of Hebrew verb inflection, has a simple structure. There are three separate lexicons: one for verb primary bases, the second for verb prefixes, and the third for verb suffixes. However, one must note that not every combination of prefix, base and suffix is indeed a meaningful lexical description. This in fact means that if we were to work along the original lines of the lexicon system, as described by Koskenniemi, we would have to split the lexicons into sub-lexicons in order to prevent over generation. We have, therefore, introduced some modifications in the way the Two Level program works with the lexicon system. Our modifications result in a simple lexicon system, in which the entries are written as normal words, without the use of non-terminal and special characters that appear on the eventual lexical string. These modifications will be described in section six.
4.5 COMPARISON WITH PREVIOUS MORPHOLOGICAL METHODS

As described in section two, the traditional description of Hebrew verb inflections was by inflection tables. The tables for the different verb patterns (Binyanim) listed all inflected forms according to tense, person, gender, and number. If these inflection tables are kept entirely in the computer memory, efficient look-up algorithms can be designed. The task of analysis using these tables is more difficult than generation, since the verb pattern cannot always be easily detected from the given inflected form, and the possibility of several correct analyses must be considered. The existence of a substantial number of classes of irregular verbs makes the storage of all the necessary tables not worthwhile.

The algorithms bases on the principles of Generative Grammar are much more efficient in space. The concept of looking at inflected forms as originating basically from the concatenation of a base and affixes requires us to store only the sets of affixes and the verb bases. The first stage of the inflection process involves converting the primary base to the secondary base, when such a secondary base exists. Then the appropriate prefixes and/or suffixes are appended, and consequently several morphological changes occur in the verb. These changes are described by rules, operating one on the result of its predecessor. The algorithm designed for analysis using this approach is more complex. This is due to problems in reversing the operations done in generation, and also to the fact, that the possibility of several correct analyses has to be taken into account.

The analysis of Hebrew verb morphology under the Two Level model uses the same concept of looking at the inflected form as a concatenation of separate lexical components. The transformation of the primary base to the secondary base is still required, and apparently cannot be described using the Two Level rules in a natural way. However, the morphological changes in the word that occur as a result of the concatenation of the affixes to the verb base are not in principle dependent one upon the other. They are local in nature, and operate on separate segments of the word. Therefore, they can generally be described by Two Level rules, which act in parallel and independently of one another. The Two Level rules serve as an entity that is separate from the system of lexicons, describing the morphological phenomena in a generalized way, similar to that of the generative grammar rules. The advantage of the Two Level model is that in principle the relationship between the lexical and surface strings is bidirectional and therefore the resulting analysis algorithm is quite similar to the generation algorithm in both program logic and complexity. However, in the few cases where there exists a dependency between two or more morphological rules, the specification of appropriate Two Level rules is more difficult, and some generalities may be lost.

5. THE PROLOG IMPLEMENTATION OF THE TWO LEVEL MODEL

5.1 MOTIVATION

The PROLOG programming language has gained popularity in the last several years as a language appropriate for Natural Language Processing. The heart of the Two Level model is the set of Two Level rules. Each rule describes a relation between a pair of characters. Combined, we can view the set of rules as actually defining a relation that holds between the Two Level strings. Therefore, our major motivation in creating a PROLOG implementation of the Two
Level model was the fact, that relations between entities can be easily specified in PROLOG. Thus, relations between pairs of characters of the two levels can be described directly by PROLOG predicates, without the need to convert the rules into finite state automata, as proposed in the original Two Level model. PROLOG also provides extensive debugging tools, that are particularly helpful in the process of developing the set of Two Level rules.

5.2 THE TWO LEVEL RULE INTERPRETER

The first component of our PROLOG implementation of the Two Level model is an interpreter for Two Level rules directly into PROLOG clauses. The interpreter, written itself in PROLOG, reads an input file, that consists of rules written in the formal syntax of Koskenniemi's Two Level rules, and creates an output file of PROLOG clauses, that retain the original relationship between lexical and surface strings, described by the Two Level rules. The Two Level program, when activated, will use these clauses to verify that a lexical and surface pair of strings complies to all Two Level rules. In fact, the program uses the clauses that describe the rules in order to generate the string of one level according to the given string of the other level. The bidirectionality of the PROLOG built in unification formalism makes the predicates which describe the Two Level rules, a common kernel of both the generation and analysis procedures.

5.3 THE LEXICON SYSTEM

The system of lexicons in our implementation is quite similar to the original system of lexicons as described by Koskenniemi in his work. A start class predicate lists the names of the lexicons that may serve as a start lexicon. Lexical strings must therefore start with an entry from one of the lexicons mentioned in this list. Each entry in a lexicon consists of a lexical entry and a list of features, indicating various information on the entry itself, such as the category of the word (verb, noun, preposition, etc.). Unlike in the original model, the feature list has an important role in the use of the lexicon system by the Two Level program, as will be described in section six. In the original lexicon system, each entry of a lexicon included a continuation class, the set of lexicons that contain valid continuations of the lexical string. In our proposed lexicon system, there is a single continuation class attached to each lexicon, and not to each entry in the lexicon. Therefore, an entry of a lexicon may continue with any entry that appears in a lexicon specified in the continuation class. It is by use of the information obtained from the features connected to the lexicon entries, that only meaningful lexical strings are eventually produced by the Two Level program.

5.4 THE TWO LEVEL PROGRAM

The program that is responsible for the tasks of generation and analysis follows the same concepts as outlined in the Two Level model. There are two major procedures in the program, one that performs generation and the other for analysis. The two are quite similar in principle. During analysis, the operations used in generation are done in the reverse order. The clauses that describe the Two Level rules are used in both procedures, to construct the string of one level from the other. However, in the course of analysis, at each stage, the partial lexical string obtained so far is checked to be meaningful, by a search through the lexicon system, to avoid the construction of non-meaningful lexical strings. The complexity of the analysis task is affected by the fact that
a given surface string may have more than a single correct analysis. Furthermore, the possibility that null characters must be inserted into the surface string at various stages, also makes the analysis task more difficult than generation.

The input to the generation procedure is a feature list that contains a base word and a description of the desired inflection (tense, person, number etc.). From this feature list the lexical string is constructed by traveling through the lexicon system and concatenating the entries that fit the features described in the input. The surface string is obtained character by character by using the clauses that implement the Two Level rules. The generated word is then obtained by deleting the null characters from the surface string.

The analysis procedure starts immediately with the construction of the lexical string from the surface string provided. The possibility of inserting a null character is examined at each stage, and the validity of the partial lexical string obtained is checked. When a lexical string is fully constructed, it is converted to the form of a base word and a feature list. The possibility of further correct analyses is checked.

6. PROPOSED EXTENSIONS AND MODIFICATIONS TO THE TWO LEVEL MODEL

6.1 THE CONVERSION PROCEDURE

In section four we discussed the problem of converting verb primary bases to secondary bases. The Two Level model, as it appears in Koskenniemi’s work, offers us no choice other than keeping all verb bases in the lexicon system. However, as we have seen, when a secondary base exists, the transformation between the primary base and the secondary base is simple and depends almost completely on the verb pattern and the tense. In our implementation, we propose adding a conversion procedure, that connects between the lexical entry obtained from the lexicon and the lexical entry that is attached to the lexical string. The conversion is bidirectional, and transforms a lexicon entry to a lexical substring according to the feature list, given as an argument to the procedure. Thus, when supplied with the tense as an element on the feature list, the conversion procedure transforms the primary base to the appropriate secondary base.

The conversion procedure is useful for more than just transforming between verb bases. One of the disadvantages of the lexicon system originally described by Koskenniemi is, that the lexicon entries have to be listed in the lexicon in the same way as they appear on the lexical string, that is including non-terminal and other special lexical characters. What this in fact means is that the lexicon entries have to be constructed with the Two Level rules in mind, and cannot be written as normal words as they appear in a dictionary, for instance. This of course makes the construction of the lexicons, or even the extending of existing lexicons, a task of an expert linguist. The conversion procedure we introduced in our implementation solves this problem. The procedure can transform normal lexical entries to the form needed on the lexical level of the Two Level strings.

Currently, in our implementation, the conversion procedure is part of the Two Level program, and it uses a preconstructed input file that indicates how lexicon entries of specified patterns and features are to be transformed. The additional processing requires additional time for analysis and generation. Another possibility would be to make the conversion procedure a preprocessor on the lexicon system itself, removing it from the Two Level program. The
procedure would then process a manually written lexicon, and create from it a new lexicon, in which the entries are of the form needed on the lexical string. This lexicon is larger than the original one, since it also contains all secondary verb bases: The Two Level program would then use the produced lexicons, rather than the original ones. The classical time versus space tradeoff exists between the possibility of incorporating the conversion procedure in the Two Level program, as opposed to making it a preprocessor.

### 6.2 THE LEXICON FEATURE HANDLING

In Koskenniemi's original lexicon system a feature list is a part of each lexicon entry. However, the features have no part in the construction of lexical paths of lexicon entries, which together build the lexical level string. The features collected from the lexicon entries that build up a lexical string are displayed to the user at the end of an analysis as a summary of the analysis results. Furthermore, the structure of the lexicon system is such that each lexical entry in a lexicon specifies its possible continuation classes. Any entry appearing in one of the lexicons specified in the continuation classes of the current lexical entry, can be appended to it. This requires that the lexicon system be split into many small lexicons, since there are various restrictions on which suffixes can be appended to a specific word base. For example, let us assume that a Hebrew base lexicon contains the following primary and secondary bases of the verb *katab*, as in figure 4.

<table>
<thead>
<tr>
<th>Entry</th>
<th>Features</th>
<th>Continuation Class</th>
</tr>
</thead>
</table>
| kAtAb | category: verb
        | tense: past   | Past_suffix_lexicon |
| kOtEb | category: verb
        | tense: present| Present_suffix_lexicon |
| kToB  | category: verb
        | tense: future, imper., infin. | Future_Impl_suf_lexicon |

**Figure 4: Bases of the verb katab.**

Note, that each of the bases has a separate continuation class, which specifies the lexicon that contains the appropriate suffixes. There is a separate lexicon for suffixes that go with each of the verb bases. If we were to put all suffixes in a single lexicon, we could construct improper lexical strings such as *kAtAb+im*, a past tense base combined with a present tense suffix.

In our implementation, we propose a modification to the structure of the lexicon system, making extensive use of the features attached to each of the lexicon entries. It is clear, that there exists a minimal number of lexicons required in order to construct lexical paths. For instance, since verb bases can accept both prefixes and suffixes, at least three lexicons are needed to create a lexical string. Similarly, if several suffixes may be concatenated one to another, they should appear in separate lexicons. However, the feature lists can handle the fact, that not every
combination of prefix, base and suffix is a valid one. A combination of such a triplet of entries is valid only if their combined features agree with each other. The continuation class is not a part of each lexicon entry, but rather belongs to the lexicon itself. The continuation class of a lexicon is the set of all possible continuation lexicons of entries in the lexicon. In our improved lexicon system, our previous example changes in the following way. We have a single suffix lexicon for all possible suffixes. Some of the entries in this lexicon are specified in figure 5.

<table>
<thead>
<tr>
<th>Entry</th>
<th>Features</th>
<th>Continuation Class</th>
</tr>
</thead>
<tbody>
<tr>
<td>ti</td>
<td>tense: past</td>
<td>End.</td>
</tr>
<tr>
<td></td>
<td>person: p1</td>
<td></td>
</tr>
<tr>
<td></td>
<td>number: sing.</td>
<td></td>
</tr>
<tr>
<td>'im</td>
<td>tense: present</td>
<td>End.</td>
</tr>
<tr>
<td></td>
<td>number: plural</td>
<td></td>
</tr>
<tr>
<td>'i</td>
<td>tense: future</td>
<td>End.</td>
</tr>
<tr>
<td></td>
<td>person: p2</td>
<td></td>
</tr>
<tr>
<td></td>
<td>gender: f</td>
<td></td>
</tr>
<tr>
<td></td>
<td>number: sing.</td>
<td></td>
</tr>
</tbody>
</table>

Figure 5: Some entries of the verb suffix lexicon.

The combination of $katAb+im$ is not possible now since the combined tense features do not agree. The system of lexicons that results from applying these principles is very structured, and can be sketched by a directed acyclic graph (DAG) of lexicons. The number of required lexicons is reduced drastically. Our modified lexicon system requires that each lexicon entry be accompanied by a feature list, that completely specifies the role of the lexical entry in the lexical string. If, for example, a certain prefix can be added to a verb base only in the future tense in the first person and plural number, these features must be specified on the feature list, attached to the lexicon entry appropriate to the prefix. On generation, the lexical string can be precisely constructed from the list of features entered by the user, by traversing through the lexicon system according to the DAG. When analysis is done, the lexical string is decomposed to obtain a precise specification of the features of the analyzed word.

7. SUMMARY

So far, our work has concentrated on two major issues. We have considered the applicability of the Two Level morphological model, first proposed by Koskenniemi in his dissertation [KOSK1], to the inflection of verbs in Hebrew. We have shown that converting between the primary and secondary base of the verb, in patterns that have several verb bases, cannot be described by Two Level rules in a natural way. If we wish to avoid keeping all possible bases in the lexicon, a conversion procedure must be added to the model, either as part of the Two Level program, or as a preprocessor. Such a conversion procedure, if added to the model, has an additional
advantage in simplifying the entries of the Two Level lexicon system. The Two Level rules can however successfully describe the changes that occur in the verb as a result of the concatenation of the affixes to the verb base. We constructed a small set of rules that describe the morphological changes that occur in verb inflections in the present tense, and are currently working on expanding this set of rules to deal with the other tenses. In the future, we would like to add to the system the handling of Hebrew nun morphology as well.

The other part of our work has been developing a new implementation in PROLOG for the Two Level morphological model. Our implementation is general and is independent of the language whose morphology is to be analyzed. We now have a quite developed prototype, and there is still work to be done until a complete version of the PROLOG implementation will be ready, mainly in making the program more efficient in terms of run time. The major advantage of our PROLOG implementation is that it converts Two Level rules, specified in formal syntax, directly into PROLOG clauses, without having to build finite state automata from the rules. There is no doubt that simulating automata in run time is inherently more efficient than verifying the Two Level rules using our PROLOG predicates. Note however, that the Two Level model itself does not imply efficient (polynomial) computational complexity, as shown by Barton [BAR1].
REFERENCES


