Arabic Computational Morphology in the West

George Anton Kiraz
Bell Laboratories
Lucent Technologies
700 Mountain Ave.
Murray Hill, NJ 07974, USA
gkiraz@research.bell-labs.com

Abstract

This paper presents an overview of research on computational Arabic morphology which is done in the West, as well as a new multi-tape model which consits of three components: A lexical component which mapps the autosegmental morphemes of Arabic (e.g., pattern, root and vocalism in stems) to a surface form. A rewrite-rules component takes care of conditional changes. A morphotactic component provides for morphotactic parsing.

1 Introduction

Research in Arabic morphology has been a popular field in the Arab World due to the importance of a morphological component in any language model of Arabic. In the West, Arabic attracts attention because of its intriguing morphological system, viz., the root-and-pattern system. However, little interaction exists between Arab and western researchers. This is apparent in works published by the two communities which lack cross-community references, understandably since most work on Arabic morphology in the Arab World is published in the native language, while western publications on the topic are usually confined to conference proceedings which are hard to access. (Finding proceedings in western libraries proofs difficult as well!) This paper is an attempt to cross this gap, albeit in one direction.

The following describes previous work on Arabic morphology from the perspective of western researchers. It merely serves as a starting point to Arab researches who are interested in the topic. ICEMCO serves as an excellent bridge where such an overview can be presented.

Section 2 introduces mainstream finite-state morphology. Section 3 outlines previous proposals and research for handling Arabic morphology. Section 4 describes a new multi-tape approach. Finally, section 5 gives concluding remarks.

2 Finite-State Morphology

The notion of using finite-state transducers (FSTs) to model phonological rules dates back to the early work of (Johnson, 1972). Independently, (Kay and Kaplan, 1983) – in an unpublished work[1] – arrived at a similar conclusion and presented the mathematical tools required for the compilation into FSTs of regular rewrite rules of the form

$$\phi \rightarrow \psi / \lambda \underline{\quad} \rho$$

where ϕ, ψ, λ and ρ are regular expressions, with ϕ designating the input, ψ designating the output, and λ and ρ designating the left and right contexts, respectively. Rules are marked with precedence (which rule applies first) and direction (whether a rule applies to the string in question left-to-right, right-to-left or simultaneously).

Each rule is compiled into a finite-state transducer. The relation between lexical and surface strings is taken as the composition of the FSTs which sanction their mapping. An analysis which involves n ordered rules requires n FSTs running on $n+1$ tapes, with $n-1$ tapes being intermediate ones. As n increases, the number of intermediate tapes increases. Though merging all FSTs into one FST is possible by composing and minimising all transducers as shown above, doing so results in huge intermediate machines which make

[1] Later published as (Kaplan and Kay, 1994).

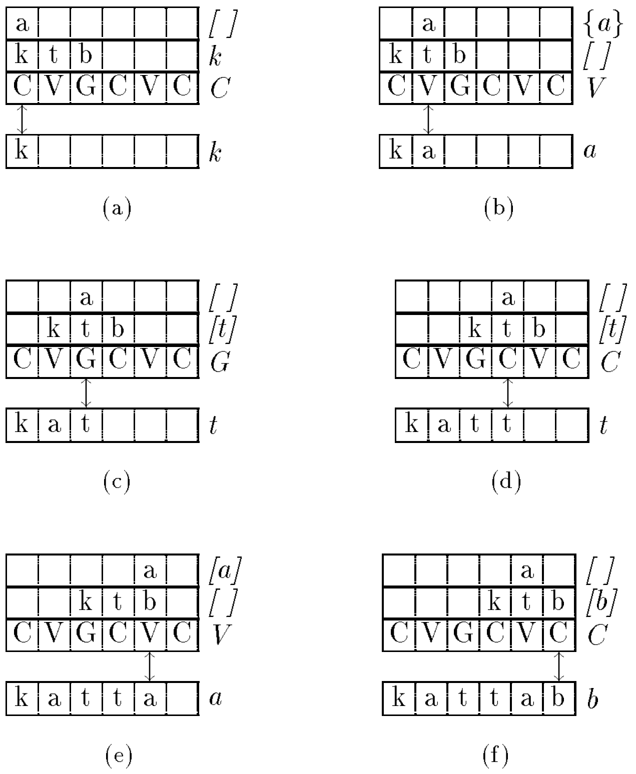
the process – at least for the computer devices of that time – computationally infeasible.

(Koskenniemi, 1983), working in the domain of morphology, proposed that the FSTs which represent rules should run in parallel, rather than serial composition. In other words, each FST must accept the **lexical representation** and the **surface representation** of the analysis. Since only two tapes are visible to all transducers, the model was named **two-level morphology**. Here, the direct relation between the lexical and surface strings is taken as the intersection of all transducers. Koskenniemi's proposal proved to be successful and was implemented in various systems (Karttunen, 1983; Antworth, 1990; Karttunen, 1993). However, it fell short of describing complex nonlinear operations.

3 Previous Proposals and Systems

A major obstacle in mainstream two-level morphology is that it assumes a liner lexical representation, making it extremely difficult, if not impossible, to apply mainstream two-level notation to the autonomous morphemes of Arabic: pattern, root and vocalism. A number of proposals for handling Arabic morphology in a more interesting way have appeared in the past decade. This section gives a brief description of them, as well as related works on other Semitic languages, in chronological order.

Figure 1: Kay's Analysis of /kattab/



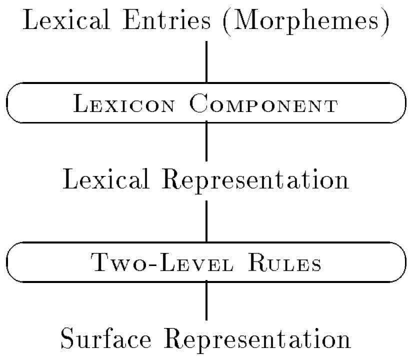
3.1 Kay's Approach to Arabic (1987)

(Kay, 1987) proposed a finite-state approach for handling Arabic nonconcatenative morphology. The approach follows the autosegmental analysis of Arabic (McCarthy, 1981). Kay's proposal uses multi-tape automata and adds some extensions to traditional FSTs. In his version, transitions are marked with quadruples of elements (for vocalism, root, pattern and surface form, respectively), where each element is a pair: a symbol and an instruction concerning the movement of the tape's head. Kay uses the following notation: An unadorned symbol is read and the tape's head moves to the next position. A symbol in brackets, [ ], is read and the tape's head remains stationary. A symbol in braces, { }, is read and the tape's head moves only if the symbol is the last one on the tape.

The transitions for the analysis of Arabic /kattab/ (Measure 2) is shown in Fig. 1. The four tapes are (from top to bottom): vocalism tape, root tape, pattern tape and surface tape. Transition quadruples are shown at the right side of tapes. The symbol '↕' between the lower surface tape and the lexical tapes indicates the current symbols under the read/write head(s). After the first transition on the quadruple ⟨[ ], k, C, k⟩, (i) no symbol is read from the vocalism tape, (ii) [k] is read from the root tape and the tape's head is moved, (iii) [C] is read from the third tape and the tape's head is moved, and (iv) [k] is written on the surface tape and the tape's head is moved. At the final configuration, all the tapes have been exhausted and the desired form appears on the surface tape.

Kay makes use of a special symbol, G, to handle gemination. When reading G on the pattern tape, the machine scans a symbol from the root tape without advancing the read head of that tape.

Figure 2: Architecture of the Akkadian System



3.2 Kataja and Koskenniemi's Approach to Akkadian (1988)

Working within standard two-level morphology, (Kataja and Koskenniemi, 1988) describe a system which handles Akkadian stems. (I shall use Arabic examples instead of Akkadian ones here.) The general architecture of the system appears in Fig. 2. Lexical entries take the following form: a verb such as /nkutib/ (Measure 7, passive) has the lexical entries

$$\Sigma_1^* \text{ k } \Sigma_1^* \text{ t } \Sigma_1^* \text{ b } \Sigma_1^*$$

and

$$\text{n } \Sigma_2 \text{ u } \Sigma_2 \text{i } \Sigma_2$$

where Σ_1 is the alphabet of the vocalism and affixes and Σ_2 is the alphabet of the root. The **lexicon component** of the system takes the intersection of the two expressions and produces the verbal stem /nkutib/ which is fed onto the lexical tape of a standard two-level system. The **two-level rules** take care of conditional phonetic changes (assimilation, deletion, etc.) and produce /ʔinkutib/ since Arabic initial consonant clusters, CC, require a prosthetic /ʔi/.

The difference between this system and a standard two-level model is the way in which the system searches the lexicon via the 'lexical component'. Instead of taking the intersection of lexical entries in advance, the system can do simultaneous searches in the lexica to simulate intersection.

3.3 Beesley's Approach to Arabic (1989, 1990, 1991)

In a number of papers, Beesley et al. report a working system for Arabic (Beesley, Buckwalter, and Newton, 1989; Beesley, 1990; Beesley, 1991). This is probably the largest reported system for Arabic morphology in the West. The lexicon contains *c.* 5000 roots which would cover all roots in Modern Standard Arabic (Wehr, 1971). Each entry is marked to show which verbal and nominal patterns a root can combine with. Another lexicon holds verbal and nominal patterns. The lexical access was named 'detouring' which simulates the intersection of two lexica.

The system was tested on newspaper texts. For each word, the system returned (1) the appropriate lexical strings with the root and pattern intersected together as a stem, (2) the root and pattern separated, (3) lists of features, and (4) a rough English meaning (Beesley, personal communication). A more recent version of this system is described by (Beesley, 1996), with an excellent demo available on the Internet (URL: `http://www.rxrc.xerox.com/research/mltt/arabic/`). We shall hear more about this in the current conference.

3.4 Lavie's Approach to Hebrew (1990)

(Lavie, Itai, and Ornan, 1990) describe a two-level system which handles Hebrew verbal stems. Their approach is similar to that of (Kataja and Koskenniemi, 1988) in that they have a lexical component which expands lexical strings. The output of this component becomes the lexical string in a standard two-level system. Since such transformations delay the analysis, they propose passing the lexicon through a preprocessor which converts primary stems into secondary ones prior to the two-level analysis.

3.5 Kornai's Linear Coding (1991)

(Kornai, 1991) proposed modeling Autosegmental Phonology using FSTs, where nonlinear autosegmental representations are coded as linear strings. Various encodings were evaluated with respect to four desiderata: computability, compositionality, invertibility and iconicity. To illustrate one encoding, consider the autosegmental representation of the Ngbaka verb /kpòlò/ 'return' in Fig. 3). The corresponding linear coding in Kornai's system is given using four keyword types as follows:

```
L0k b L0p b L1o b L1o
```

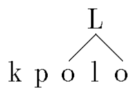
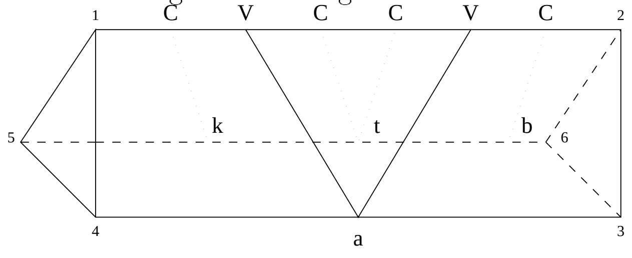


Figure 3: Autosegmental Representation of /kpòlò/

Figure 4: Triangular Prism



The coding L0k indicates the absence of an association line between [L] on the upper tone tier and [k] on the lower stem tier. The keyword b advances the bottom tape only. The next two keywords are similar to the two preceding ones, respectively. The keyword L1o indicates the existence of an association line between [L] and [o]. Kornai also uses the keyword t (which does not feature in this example) to advance the top tape only. Such linear encodings are converted into traditional finite-state transducers.

3.6 Bird and Ellison's One-Level Approach (1992, 1994)

(Bird and Ellison, 1994)[2] proposed a model based on *one*-level phonology using FSAs to model representations and rules. Their model employs an encoding scheme for the representation of autosegmental diagrams. Every pair of autosegmental tiers constitutes a **chart** (or plane). This is illustrated for the case of Arabic /kattab/ in the form of a triangular prism as in Fig. 4. Each morpheme sits on one of the prism's three longitudinal edges: the pattern on edge 1-2, the vocalism on edge 3-4, and the root on edge 5-6. Moreover, the prism has three longitudinal charts (or plane): the pattern-vocalism chart (1-2-3-4), the pattern-root chart (1-2-6-5), and the root-vocalism chart (3-4-5-6). The corresponding encoding of the diagram is

a:2:0:0
C:0:1:0 V:1:0:0 C:0:1:0 C:0:1:0 V:1:0:0 C:0:1:0

[2]The original work appeared as (Bird and Ellison, 1992).

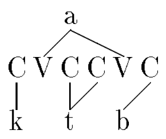


Figure 5: Autosegmental Representation of /kattab/

k:0:1:0 t:0:2:0 b:0:1:0

Each expression is an $(n+1)$-tuple, where n is the number of charts. The first element in the tuple represents the autosegment. The positions of the remaining elements in the tuple indicate the chart in which an association line occurs, and the numerals indicate the number of association lines on that chart. For example, the expression a:2:0:0 states that the autosegment a has two association lines on the first (i.e. pattern-vocalism) chart, zero lines on the second (i.e. pattern-root) chart and zero lines on the third (i.e. root-vocalism) chart.

(Bird and Ellison, 1994) provide tools for converting such encodings in finite-state devices which they call 'state-labeled finite automata'. Their machines are no more expressive than traditional FSAs.[3]

Their one-level analysis of Arabic takes the following form: three SFAs represent a pattern morpheme, a root morpheme and a vocalism morpheme, respectively. The surface form is obtained by taking the intersection of the three automata.

3.7 Wiebe's Multi-Linear Coding (1992)

(Wiebe, 1992) proposed another encoding for representing Autosegmental Phonology following Kornai's four desiderata (see §3.5). This is illustrated by showing the encoding of the autosegmental representation of Arabic /kattab/ (Measure 1). The autosegmental representation is given in Fig. 5. The corresponding 'multi-linear' coding is

a11
C2V1C2C2V1C2
k2t22b2

A numeral n following an autosegment indicate that it has an association on chart n. An autosegment which is linked m times is followed by m repetitions of n, e.g. a11 and t22.

[3]These machines are identical to Moore machines (Moore, 1956).

The multi-linear encoding is processed by devices which Wiebe calls 'multi-tape state-labeled finite automata'.[4] Labels here are associated with states rather than transitions. The computational power of Wiebe's version of these machines exceeds the power of conventional transducers. Wiebe's machines can accept context-free, and even some context-sensitive, languages.

3.8 Pulman and Hepple's Approach to Arabic (1993)

(Pulman and Hepple, 1993) proposed a formalism (which we adopt in section 4.2) for bidirectional segmental phonological processing and proposed using it for Arabic in the following manner: A stem like /takattab/ (Measure 5) is simply expressed with the rule

$$\begin{array}{ccccccc} * & - & C_1\ C_2\ C_3 & - & * & \Rightarrow \\ * & - & \text{t a } C_1 \text{ a } C_2\ C_2 \text{ a } C_3 & - & * & \end{array}$$

where C_i represents the ith radical of the root {ktb}. Note that the pattern and vocalism morphemes are embedded in the surface expression of the rule.

3.9 Narayanan and Hashem's Three-Level Approach (1993)

(Narayanan and Hashem, 1993) proposed an extension to the traditional two-level model by adding a third abstract level of automaton representation "at which classes of inflectional phenomena are given an abstract representation." Under this framework, patterns of inflection constitute an abstract automaton component which sits on top of a standard two-level system.

Their Arabic implementation assumes that nominal forms are entered in the lexicon as stems. The automata represent various nominal inflections (prefixation and suffixation), which are in fact concatenative. The treatment of the verbal system employs two-way FSTs with ε moves, mapping unequal sequences of symbols.

4 A Multi-Tape Model

This section describes a new model for Arabic morphology based on earlier reports by (Kiraz,

[4]The machines in (Bird and Ellison, 1994) differ in definition and computational power from those of (Wiebe, 1992) though both have a somewhat similar name. (Wiebe, 1992) borrowed the name from (Bird and Ellison, 1992).

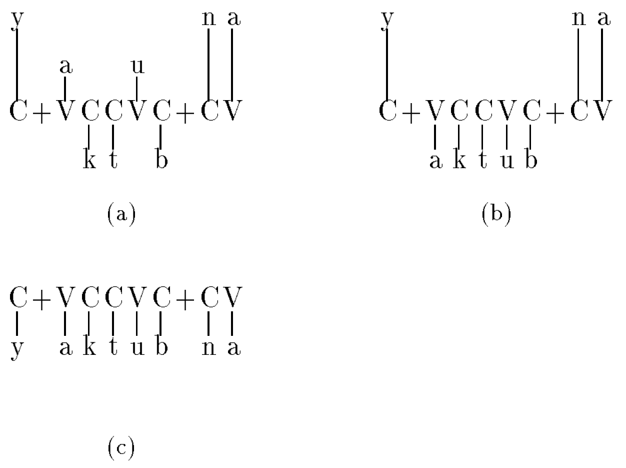


Figure 6: Autosegmental Representation of /yaktubna/

1994, et. seq.) (for the latest account, see (Kiraz, Forthcoming)).

In the case of Arabic, the lexical description consists of three lexical morphemes: pattern, root and vocalism. The model presented here is motivated by Kay's proposal (see §3.1) in that it uses multi-tape automata. It follows traditional two-level morphology, however, in assuming *two* **levels** of linguistic description in recognition and synthesis. The difference here lies in the number of **representations** each level may have: the lexical level employs *multiple* representations, while the surface level employs only *one* representation. The upper bound on the number of lexical representations is not only language-specific, but also grammar-specific. Note that a distinction is made here between the terms 'levels' and 'representations': a level may have multiple representations.

There is a linguistic motivation behind this model, As described by (McCarthy, 1986), when a word is uttered, it is pronounced in a linear string of segments, i.e. the multi-tier lexical representation is linearised at the surface level. McCarthy calls this process **tier conflation**. Consider for example the autosegmental structure in Fig. 6(a) of Arabic /yaktubna/ 'they (FEM) are writing' which consists of the following morphemes (from top to bottom): the circumfix morpheme {y-na} 'IMPF PL 3RD FEM', the vocalism morpheme {au} 'IMPF ACT', the corresponding CV pattern morpheme and the root morpheme {ktb} 'notion of writing'. Tier conflation is performed as follows: after associating the root consonants with the Cs and the vocalism vowels with the Vs, one 'folds together' the consonants and vowels of the stem onto a single tier as shown in Fig. 6(b). The same operation is performed on the remaining morphologically determined tiers resulting in the linearised configuration in Fig. 6(c). In other words, the various lexical tiers in the underlying representation end up in a single tier in the surface level.

In this vein, the lexicon component of the new model consists of multiple sublexica, each representing entries for a particular lexical representation or tier. The rewrite rules component maps the multiple lexical representations to a surface representation. Finally, the morphotactic component attempts to find a morphotactic parse for the lexical forms in questions. Here, the input to the morphotactic parser is a tuple of lexical forms, where each element in the tuple represents the lexical forms from one of the sublexica.

4.1 The Lexicon Component

The lexicon in the current model consists of multiple sublexica, each sublexicon containing entries for one particular lexical representation (or tier in the autosegmental analysis). Since an n-tuple contains $n-1$ lexical elements, the lexicon component consists of $n-1$ sublexica. An Arabic lexicon, for example, will have a pattern sublexicon, a root sublexicon and a vocalism sublexicon. Other affixes which do not conform to the root-and-pattern nature of Semitic morphology (e.g. prefixes, suffixes, particles, etc.) need to be represented as well. One can either give them their own sublexicon or have them represented in one of the three sublexica. Since pattern segments are the closest in terms of number to surface segments, we chose the convention of having such morphemes represented in the pattern sublexicon.

For morphotactic purposes, each entry in a sublexicon is associated with a **category-feature structure** of the form,

$$\begin{bmatrix} \textsf{cat} \\ \text{ATTRIBUTE}_1 = \text{value}_1 \\ \text{ATTRIBUTE}_2 = \text{value}_2 \\ \vdots \end{bmatrix}$$

where *cat* is an atom representing a (grammatical) category followed by an unordered list of *attribute=value* pairs. An *attribute* is an atomic label. A *value* can be an atom or a variable drawn from a predefined finite set of possible values.[5]

As a way of illustration, consider the Arabic verb /katab/ with the conjunction prefix {wa} 'and' and the suffix {at} 'SING 2ND FEM'. The entries of the first sublexicon are

$$\text{cvcvc} \begin{bmatrix} \textsf{pattern} \\ \text{MEASURE} = 1 \\ \text{VOICE} = \text{act} \end{bmatrix} \qquad \text{wa} \begin{bmatrix} \textsf{conj} \end{bmatrix}$$

$$\text{at} \begin{bmatrix} \textsf{vim} \\ \text{NUMBER} = \text{sing} \\ \text{PERSON} = \text{2nd} \\ \text{GENDER} = \text{fem} \end{bmatrix}$$

The second sublexicon maintains the root entry

$$\text{ktb} \begin{bmatrix} \textsf{pattern} \end{bmatrix}$$

while the third lexicon maintains the vocalism

$$\text{a} \begin{bmatrix} \textsf{vocalism} \end{bmatrix}$$

In a working system, other patterns, roots and vocalisms will be maintained in the respective sublexica.

4.2 The Rewrite Rules Component

The rewrite rules component maps the multiple lexical representations to a surface representation. It also provides for phonological, orthographic, etc., rules. As before, n-tuples are used for lexical-surface mapping, with the first element representing surface forms and the remaining $n-1$ elements representing lexical forms.

Rewrite rules are usually expressed in some formalism. The current model adopts the following formalism:[6]

$$\begin{array}{lclcll} \text{LLC} & - & \textsc{Lex} & - & \text{RLC} & \{\Rightarrow, \Leftrightarrow\} \\ \text{LSC} & - & \textsc{Surf} & - & \text{RSC} & \end{array}$$

LLC denotes the left lexical context, LEX denotes the lexical form, and RLC denotes the right lexical context. LSC, SURF and RSC are the surface counterparts. The context denoted by '*' is always satisfied, i.e. it represents Kleene star.

[5] It is also possible to extend the above formalism in order to allow *value* to be a category-feature structure, though this takes us beyond finite-state power.

[6] For the historical development of this formalism, see (Kiraz, Forthcoming).

Figure 7: Derivation of /katab/

```
     *   –  ⟨c,X,ε⟩  –  *  ⇒
R1:  *   –     X     –  *
     where X is a consonant.

     *   –  ⟨v,ε,X⟩  –  *  ⇒
R2:  *   –     X     –  *
     where X is a vowel.

     ⟨v,ε,X⟩ *  –  v  –  *  ⇒
R3:      *      –  X  –  *
     where X is a vowel.
```

	a				*vocalism*
k		t		b	*root*
c	v	c	v	c	*pattern*
1	2	1	3	1	
k	a	t	a	b	*surface*

(a)

			a						*vocalism*
		k		t		b			*root*
w	a	c	v	c	v	c	a	t	*pattern & affixes*
0	0	1	2	1	3	1	0	0	
w	a	k	a	t	a	b	a	t	*surface*

(b)

Figure 8: Analysis of /katab/ and /wakatabat/

Capital-initial expressions are variables over predefined finite sets of symbols.

The operator ⇒ is the context restriction operator. It states that Lex *may* surface as Surf in the given context. The operator ⇔ adds surface coercion constraints: when Lex appears in the given context, then the surface description *must* satisfy Surf. A lexical string maps to a surface string if and only if they can be partitioned into pairs of lexical-surface subsequences, where (1) each pair is licensed by a ⇒ rule, and (2) no sequence of zero or more adjacent pairs violates a ⇔ rule.

For Arabic, all expressions on the upper lexical side of the rules (LLC, Lex and RLC) are tuples of strings of the form $\langle x_1, x_2, \ldots, x_{n-1} \rangle$. The ith element in the tuple refers to symbols in the ith sublexicon of the lexical component. When a lexical expression makes use of only the first sublexicon, the angle brackets can be ignored. Hence, the Lex expression $\langle x, \epsilon, \ldots, \epsilon \rangle$ and x are equivalent; in lexical contexts, $\langle x, *, \ldots, * \rangle$ and x are equivalent.

The formalism is illustrated in Fig. 7. The rules derive Arabic /katab/ from the pattern morpheme {cvcvc} 'verbal Measure 1', the root morpheme {ktb} 'notion of writing' and the vocalism morpheme {a} 'perf act'. R1 sanctions root consonants by mapping a [c] from the first (pattern) sublexicon, a consonant [X] from the second (root) sublexicon and no symbol from the third (vocalism) sublexicon to surface [X]. R2 sanctions vowels in a similar manner. R3 allows the spreading of the vowel: if a vowel [X] has previously occurred, i.e. LLC is '⟨v,ε,X⟩ *', then a [v] from the pattern sublexicon may map to that same vowel on the surface. The quality of the vowel on the surface is determined by the unification of X in Surf and LLC.

The mapping is illustrated in Fig. 8(a). The numbers between the surface and lexical expressions indicate the rules in Fig. 7 which sanction the shown subsequences. Empty slots represent the empty string ϵ. Note that we depict expressions from bottom to top: first the surface expressions, then the lexical expressions.

In the lexicon, morphemes which do not conform to the root-and-pattern nature of Semitic (i.e. prefixes, suffixes, particles, etc.) are given in the first sublexicon. The identity rule

```
     *  –  X  –  *  ⇒
R0   *  –  X  –  *
     where X∉ { c,v }
```

maps such morphemes to the surface. The rule basically states that any symbol not in { c,v } from the first sublexicon may surface. Fig. 8(b) illustrates the analysis of /wakatabat/.

The rewrite rule component interacts with the lexical component in the following manner. The lexical expressions produced by a rewrite rule must each represent a concatenation of lexical entries from the corresponding sublexicon. For example, for the analysis described by the tuple ⟨wakatabat, wa cvcvc at, ktb, a⟩ to be lexically valid, the first sublexicon must contain the entries {wa}, {cvcvc} and {at}. Similarly, the second and third sublexica must contain the entries {ktb} and {a}, respectively.

4.3 The Morphotactic Component

A two-level grammar and a lexicon fall short of defining the set of licit combinations of lexical forms. In computational morphology, there are two schools of thought in this regard. The first uses 'continuation patterns/classes' (Koskenniemi, 1983; Antworth, 1990; Karttunen, 1993) where each class of morphemes is associated with a set of continuation classes. The second adopts unification-based grammars with linear precedence relations (Bear, 1986; Beesley, Buckwalter, and Newton, 1989; Trost, 1990; Ritchie et al., 1992; Antworth, 1994, *inter alia*). The former fails to provide elegant morphotactic parsing for Arabic. The latter provides facilities to describe more complex morphotactic rules, but still needs to be modified in order to cope with Semitic stems.

The current morphotactic component takes as input the (grammatical) categories – in the form of category-feature structures – of the morphemes realised from the lexicon and rewrite rules components. Since, the lexical and rewrite rules components produce a tuple of sequences of lexical forms, one per lexical expression, we use context-free relations (i.e., context-free grammars with terminals being tuples of symbols) to describe morphotactics. The analysis of Arabic /wakatabat/ produces the lexical 3-tuple ⟨wa cvcvc at, ktb, a⟩, i.e. the sequence of morphemes on each lexical element. The corresponding grammatical categories of these morphemes (based on the entries from section 4.1) are

$$\left\langle \begin{bmatrix}\textsf{conj}\end{bmatrix} \begin{bmatrix}\textsf{pattern}\\ \text{MEASURE} = 1\\ \text{VOICE} = \text{act}\end{bmatrix} \begin{bmatrix}\textsf{vim}\\ \text{NUMBER} = \text{sing}\\ \text{PERSON} = \text{2nd}\\ \text{GENDER} = \text{fem}\end{bmatrix}, \begin{bmatrix}\textsf{root}\end{bmatrix}, \begin{bmatrix}\textsf{vocalism}\end{bmatrix} \right\rangle$$

Since there is linear precedence among the daughters of the derived word, one can express morphotactics in the following productions:

$$\begin{bmatrix}\textsf{Word}\end{bmatrix} \rightarrow \begin{bmatrix}\textsf{conj}\end{bmatrix} \begin{bmatrix}\textsf{Stem}\end{bmatrix} \begin{bmatrix}\textsf{vim}\\ \text{NUMBER} = \text{sing}\\ \text{PERSON} = \text{2nd}\\ \text{GENDER} = \text{fem}\end{bmatrix}$$

The derivation of $\begin{bmatrix}\textsf{Stem}\end{bmatrix}$ from the pattern ({cvcvc}), root ({ktb}) and vocalism ({a}) morphemes are described in the production:

$$\begin{bmatrix}\textsf{Stem}\end{bmatrix} \rightarrow \left\langle \begin{bmatrix}\textsf{pattern}\\ \text{MEASURE} = 1\\ \text{VOICE} = \text{act}\end{bmatrix}, \begin{bmatrix}\textsf{root}\end{bmatrix}, \begin{bmatrix}\textsf{vocalism}\end{bmatrix} \right\rangle$$

where terminals are tuples of symbols.

5 Conclusion

Arabic computational morphology is still lacking in a number of areas. None of the above works, for example, look into morphological ambiguity, a notorious problem in all Semitic languages since the orthographic representation of words lacks short vowels, case endings and other markers.

It is hoped that this paper gave our fellow researchers in the Arab World a flavor of the activities in the West on this topic. It is hoped that the research activities which take place in the Arab World will be reported for the Western researcher in the future.

References

Antworth, E. 1990. *PC-KIMMO: A two-Level Processor for Morphological Analysis.* Occasional Publications in Academic Computing 16. Summer Institute of Linguistics, Dallas.

Antworth, E. 1994. Morphological parsing with a unification-based word grammar. In *North Texas Natural Language Processing Workshop.* [http://www.sil.org/pckimmo/ntnlp94.html].

Bear, J. 1986. A morphological recognizer with syntactic and phonological rules. In *COLING-86: Papers Presented to the 11th International Conference on Computational Linguistics*, pages 272–6.

Beesley, K. 1990. Finite-state description of Arabic morphology. In *Proceedings of the Second Cambridge Conference: Bilingual Computing in Arabic and English.*

Beesley, K. 1991. Computer analysis of Arabic morphology. In B. Comrie and M. Eid, editors, *Perspectives on Arabic Linguistics III: Papers from the Third Annual Symposium on Arabic Linguistics.* Benjamins, Amsterdam.

Beesley, K. 1996. Arabic finite-state morphological analysis and generation. In *COLING-96: Papers Presented to the 16th International Conference on Computational Linguistics.*

Beesley, K., T. Buckwalter, and S. Newton. 1989. Two-level finite-state analysis of Arabic morphology. In *Proceedings of the Seminar on Bilingual Computing in Arabic and English.* The Literary and Linguistic Computing Centre, Cambridge.

Bird, S. and T. Ellison. 1992. One-level phonology: Autosegmental representations and rules as finite-state automata. Technical report, University of Edinburgh, Research Paper EUCCS/RP-51.

Bird, S. and T. Ellison. 1994. One-level phonology. *Computational Linguistics*, 20(1):55–90.

Johnson, C. 1972. *Formal Aspects of Phonological Description*. Mouton.

Kaplan, R. and M. Kay. 1994. Regular models of phonological rule systems. *Computational Linguistics*, 20(3):331–78.

Karttunen, L. 1983. Kimmo: A general morphological processor. *Texas Linguistic Forum*, 22:165–86.

Karttunen, L. 1993. Finite-state lexicon compiler. Technical report, Palo Alto Research Center, Xerox Corporation.

Kataja, L. and K. Koskenniemi. 1988. Finite state description of Semitic morphology. In *COLING-88: Papers Presented to the 12th International Conference on Computational Linguistics*, volume 1, pages 313–15.

Kay, M. 1987. Nonconcatenative finite-state morphology. In *Proceedings of the Third Conference of the European Chapter of the Association for Computational Linguistics*, pages 2–10.

Kay, M. and R. Kaplan. 1983. Word recognition. This paper was never published. The core ideas are published in Kaplan and Kay (1994).

Kiraz, G. 1994. Multi-tape two-level morphology: a case study in Semitic non-linear morphology. In *COLING-94: Papers Presented to the 15th International Conference on Computational Linguistics*, volume 1, pages 180–6.

Kiraz, G. [Forthcoming]. *Computational Approach to Nonlinear Morphology: with emphasis on Semitic languages.* Cambridge University Press.

Kornai, A. 1991. *Formal Phonology*. Ph.D. thesis, Stanford University. Later published as (Kornai, 1995).

Koskenniemi, K. 1983. *Two-Level Morphology*. Ph.D. thesis, University of Helsinki.

Lavie, A., A. Itai, and U. Ornan. 1990. On the applicability of two level morphology to the inflection of Hebrew verbs. In Y. Choueka, editor, *Literary and Linguistic Computing 1988: Proceedings of the 15th International Conference*, pages 246–60.

McCarthy, J. 1981. A prosodic theory of nonconcatenative morphology. *Linguistic Inquiry*, 12(3):373–418.

McCarthy, J. 1986. OCP effects: gemination and antigemination. *Linguistic Inquiry*, 17.

Moore, E. 1956. Gedanken experiments on sequential machines. In *Automata Studies*. Princeton University Press, pages 129–53.

Narayanan, A. and L. Hashem. 1993. On abstract finite-state morphology. In *Proceedings of the Sixth Conference of the European Chapter of the Association for Computational Linguistics*, pages 297–304.

Pulman, S. and M. Hepple. 1993. A feature-based formalism for two-level phonology: a description and implementation. *Computer Speech and Language*, 7:333–58.

Ritchie, G., A. Black, G. Russell, and S. Pulman. 1992. *Computational Morphology: Practical Mechanisms for the English Lexicon.* MIT Press, Cambridge Mass.

Trost, H. 1990. The application of two-level morphology to non-concatenative German morphology. In H. Karlgren, editor, *COLING-90: Papers Presented to the 13th International Conference on Computational Linguistics*, volume 2, pages 371–6.

Wehr, H. 1971. *A Dictionary of Modern Written Arabic*. Spoken Language Services, Ithaca.

Wiebe, B. 1992. Modelling autosegmental phonology with multi-tape finite state transducers. Master's thesis, Simon Fraser University.