Linearization of Nonlinear Lexical Representations

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

Abstract

This paper presents a new schema for handling nonlinear morphology. The schema argues for linearizing nonlinear representations before applying phonological and morphological rules.

1 Introduction and Problem Statement

Languages which exhibit templatic morphology have been lately treated using multi-tape finite state transducers, with one tape representing surface forms and the remaining tapes representing lexical forms (see (Kay, 1987; Kiraz, Forthcoming)). There are a number of advantages for using this multi-tape model. Not only does it accurately represent the linguistic insights behind the templatic nonlinear nature of these languages, it also allows the computational linguist to compile efficient, relatively small morphological lexica as opposed to lexica containing millions of entries.

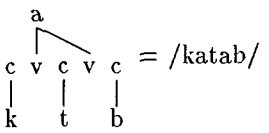
However, maintaining a nonlinear lexical representation has its own inconveniences and computational complexities. Firstly, the writer of multi-tape rules must keep track of multiple representations (four in the case of Semitic as opposed to two for English), which makes writing grammars an arduous task. Secondly, rules which describe one phonological/orthographic phenomenon must be duplicated in order to account for the nonlinear nature of the stem, but the linear nature of segments present in prefixes and suffixes. Thirdly, in systems which require multiple sets of rules (say a text-to-phoneme system with two sets of rules: surface-to-lexical and lexical-to-phoneme), the above complexities multiply. Finally, there is the issue of space complexity: although the space complexity for transitions of an automata with respect to the number of tapes is linear, space can become costly for huge machines, especially those whose number of transitions exceeds by far the number of states, a typical situation in natural language problems.

This paper provides a finite-state schema with which one can maintain the nonlinear lexical representation in templatic morphology, yet allow for a linear model for representing phonological/orthographic and other script related rules. Such rules are in fact linear and need not be made complex on the account of the nonlinear templatic phenomenon of morphology.

2 Problems in Templatic Morphology

2.1 Nonlinearity vs. Linearity

Consider the infamous Arabic stem /katab/ 'to write – PERFECT ACTIVE'. It is derived from the root morpheme {ktb} 'notion of writing', the vocalism morpheme {a} 'PERFECT ACTIVE' and the rather abstract pattern morpheme {CVCVC} 'VERB.' The latter describes the interdigination of the root and vocalism. Substituting the Cs with the root consonants and the Vs with the vocalism vowels results in the surface form /katab/. This process is illustrated along the lines of (McCarthy, 1981) – based on autosegmental phonology (Goldsmith, 1976) as follows:



= /katab/

Similarly, applying the same process on the root {ṣdq} 'notion of truth' results in the verb /ṣadaq/ 'to speak the truth – PERFECT ACTIVE'.

The stems /katab/ and /ṣadaq/ may be prefixed and suffixed to form other words. Prefixation and suffixation, however, are linear operations in Semitic. In other words, the lexical representation of the prefixes and suffixes does not require multi-

ple tapes. Hence, the prefix {wa} 'and' is applied to the above stems to form /wakatab/ and /waṣadaq/, respectively.

2.2 Phonological and Orthographic Rules

Surface-to-lexical mappings must account for phonological and orthographic processes. In fact, for many languages, the phonological and orthographic rules tend to be more numerous than the morphological rules. This is the case in Semitic. For example, the Syriac grammar reported in (Kiraz, 1996) contains 48 rules. Only six rules (a mere 12.5%)[1] are motivated by templatic morphology. The rest are phonological and orthographic.

Consider the above derivation of /katab/, but for Syriac rather than Arabic (both languages share the same morphemes in this case). Syriac has the Vowel Deletion Rule

$$V \rightarrow \epsilon / \_\_ CV$$

where ϵ is the empty string. The rule states that *short* vowels in open syllables are deleted. Hence, */katab/ → /ktab/. The rule applies right-to-left; hence, when adding the object pronominal suffix {eh} 'MASCULINE 3RD SINGULAR', the second vowel is deleted, */katabeh/ → /katbeh/.

Similarly, prefixing the above {wa} morpheme (which is also shared by Syriac and Arabic), results in */wakatab/ → /waktab/ (first stem vowel is deleted), and */wakatabeh/ → /wkatbeh/ (prefix vowel and second stem vowel are deleted).

It is worth noting that such phonological rules do not depend on the nonlinear lexical structure of the stem. They actually apply on the morphologically derived stem. Semitic, then, maintains at least the following strata: lexical-morphological (where the lexical representation is nonlinear) and morphological-surface (where both representations are linear).

2.3 Other Linguistic Representations

So far we have looked at two linguistic representations: lexical and surface (≈ orthographic). Now consider a text-to-speech system which requires a phonological representation as well.

In the Arabic example above, the first phoneme of /ṣadaq/ is emphatic (denoted by the sublinear dot). This emphasis is spread at the phonological level resulting in [ṣaḍaq] ([q] is already an emphatic phoneme).[2] In this case, emphasis can be determined from the surface (≈ orthographic) form. However, this is not always the case. Syriac spirantization requires lexical information as the following example illustrates: Synchronically speaking, the six plosives [b], [g], [d], [k], [p] and [t] undergo spirantization when in postvocalic position *with respect to the lexical form*,[3] resulting in [v], [ğ], [ð], [x], [f] and [θ], respectively. Hence, */katab/ → [kθav], and */wakatab/ → [waxθav] (in both cases the first stem vowel is deleted as described above).

3 Multi-Tape Grammar

This section provides a grammar for the above data using a multi-tape model and illustrates some of the complexities involved in maintaining multiple lexical tapes throughout. The multi-tape model (originally proposed by (Kay, 1987)) is an extension to the commonly used regular rewrite rules. In the multi-tape version, more than one lexical tape is allowed. Here, we shall use the following formalism – which derives from the one reported by (Pulman and Hepple, 1993) – to express regular rewrite rules:

LLC – LEX – RLC $\{\Rightarrow, \Leftrightarrow\}$
LSC – SURF – RSC

where LLC is the left lexical context, LEX is the lexical form, RLC is the right lexical context, LSC is the left surface context, SURF is the surface form, and RSC is the right surface context. The operators $\Rightarrow$ and $\Leftrightarrow$ indicate optional and obligatory rules, respectively. In the multi-tape version, lexical expressions are n-tuple of regular expressions of the form $(x_1, x_2, \ldots, x_n)$, with the ith expression referring to symbols on the ith lexical tape. When $n = 1$, the parentheses can be ignored; hence, (x) and x are equivalent.[4]

The grammars presented here assumes a lexicon with the morpheme entries presented above. The pattern morpheme is {cvcvc} (in small letters); capitals in rules denote variables drawn from a finite-set of symbols.

Lexical expressions make use of three tapes: pattern, root and vocalism, respectively. Hence, the

[1] Had the grammar been more exhaustive, the percentage would be much less since most additions to the rules would be in the domain of phonology/orthography, rather than templatic morphology.

[2] The scope of emphasis is another challenging problem. Sometimes emphasis spreads till the end of the current syllable, and sometimes till the end of the word.

[3] Diachronically speaking, early Aramaic idioms, of which Syriac is one, did not apply the above vowel deletion rule; hence, in the New Testament the first [a] in *sabachthani* (Mt 27:46) is retained. Later, however, the vowel deletion rule took effect, but spirantized consonants remained as if the deletion did not take place.

[4] For compiling such rules into automata, see (Grimley-Evans, Kiraz, and Pulman, 1996).

Grammar 1 Grammar for Arabic /(wa)katab/ and /(wa)ṣadaq/

R1	*	–	⟨c,C,ε⟩	–	*	⇒	
	*	–	C	–	*		
R2	*	–	⟨v,ε,a⟩	–	*	⇒	
	*	–	a	–	*		
R3	*	–	X	–	*	⇒	
	*	–	X	–	*		

where X is any segment, C is a consonant, and * is any context.

lexical expression ⟨c,k,ε⟩ denotes a [c] on the first (pattern) tape, a [k] on the second (root) tape and the empty string on the third (vocalism) tape. Prefixation and suffixation, which for the most part fall out of the domain of templatic morphology, are represented as a sequence of segments as in any other language and are placed on the first (pattern) lexical tape.[5]

3.1 Nonlinearity vs. Linearity

Rules R1 and R2 in Grammar 1 take care of consonants and vowels, respectively. The rules derive the Arabic forms /katab/ and /ṣadaq/. R3 is the default rule for prefixes and suffixes. It simply maps every segment on the *first* lexical tape to the surface. Grammar 1 derives the forms /wakatab/ and /waṣadaq/ as well. The former is illustrated below.

			a		a		*Vocalism*
		k		t		b	*Root*
w	a	c	v	c	v	c	*Pattern and Affixes*
3	3	1	2	1	2	1	
w	a	k	a	t	a	b	*Surface*

The numbers between the tapes refer to the rules in the grammar. Note that the prefix shares a tape with the pattern.

3.2 Phonological and Orthographic Rules

The Syriac vowel deletion rule, V → ϵ/___ CV, is given in the notation of our formalism in Grammar 2. Note that by virtue of its right-lexical context ⟨cv,C,a⟩, R4 can only apply to the *first* stem vowel as illustrated in the derivation of /ktab/ from the underlying */katab/ by the deletion of the first vowel:

[5]Having the prefixes share a tape of the patterns is a matter of convenience since the number of segments in a pattern, more or less, corresponds to that on the surface more than segments of roots and vocalisms.

Grammar 2 Grammar for Syriac Vowel Deletion Rule

R4	*	–	⟨v,ε,a⟩	–	⟨cv,C,a⟩	⇔	
	*	–		–	*		
R5	*	–	⟨v,ε,a⟩	–	⟨cV,C,ε⟩	⇔	
	*	–		–	*		
R6	*	–	a	–	⟨cv,C,a⟩	⇔	
	*	–		–	*		
R7	*	–	a	–	CV	⇔	
	*	–		–	*		

where C is a consonant and V is a vowel

	a		a		*Vocalism*
k		t		b	*Root*
c	v	c	v	c	*Pattern*
1	4	1	2	1	
k		t	a	b	*Surface*

Another rule (R5 in Grammar 2) is required for deriving */katab/ + {eh} → /katbeh/, where the *second* stem vowel is deleted by the same phonological phenomenon. The difference here lies in the right-lexical context expression ⟨cV,C,ε⟩, where the suffix vowel appears on the first lexical tape. The derivation is illustrated below:

	a		a				*Vocalism*
k		t		b			*Root*
c	v	c	v	c	e	h	*Pattern and Affixes*
1	2	1	5	1	3	3	
k	a	t		b	e	h	*Surface*

R4 and R5 fail when the deleted vowel itself appears in the prefix, e.g. {wa} + /katbeh/ → /wkatbeh/. R6 handles this case; here, the right context ⟨cv,C,a⟩ belongs to the nonlinear stem as shown below:

			a		a				*Vocalism*
		k		t		b			*Root*
w	a	c	v	c	v	c	e	h	*Pattern and Affixes*
3	6	1	2	1	5	1	3	3	
w		k	a	t		b	e	h	*Surface*

In addition, R7 deletes prefix vowels when the right context belongs to a (possibly another) linear prefix, e.g., {wa} + {la} + {da} + /katab/ → /waldaktab/ (the [a] of {la} and the first stem vowel are deleted), as illustrated below:

							a		a		*Vocalism*
						k		t		b	*Root*
w	a	l	a	d	a	c	v	c	v	c	*Pattern and Affixes*
3	3	3	7	3	3	1	4	1	2	1	
w	a	l		d	a	k		t	a	b	*Surface*

The above examples clearly illustrate the complexity of maintaining large nonlinear grammars.

4 Using a Linearized Lexical Representation

This section argues that a better framework for solving Semitic morphology divides the lexical-surface mappings into two separate problems. The first handles the templatic nature of morphology, mapping the multiple lexical representation into a **linearized lexical form**. This linearized form maintains the same linguistic information of the original lexical representation, and somewhat corresponds to McCarthy's notion of **tier conflation** (McCarthy, 1986).

The second takes care of phonological/ orthographic/graphemic mappings between the linearized lexical form and the actual surface. The combined machine is mathematically taken as the composition of the two machines representing the two sets of rules. This brings us to the question of composing multi-tape automata.

4.1 Composition of Multi-Tape Machines

The composition of two binary transducers A and B is straightforward since one tape is taken for input and the other for output. The composition of the two machines is a generalization of the intersection of the same two automata in that each state in the resulting machine is a pair drawn from one state in A and the other from B, and each transition corresponds to a pair of transitions, one from A and the other from B, with compatible labels.

The composition of multi-tape transducers, however, is ambiguous. Which tapes are input and which are output? Consider the machine which accepts the regular relation[6] $a^*{:}b^*{:}b^*$ and a second machine which accepts the regular relation $b^*{:}b^*{:}c^*$. The composition of the two machines can be either the machine accepting $a^*{:}c^*$ or the machine accepting $a^*{:}b^*{:}b^*{:}c^*$. However, if tapes can be marked as belonging to the domain or range of the transduction, the ambiguity will be resolved.

Formally, an **n-tape finite-state automaton** is a 5-tuple $M = (Q, \Sigma, \delta, q_0, F)$, where Q is a finite set of **states**, Σ is a finite **input alphabet** (a set of n-tuples of symbols), δ is a **transition function** mapping $Q \times \Sigma^n$ to Q, $q_0 \in Q$ is an **initial state**, and $F \subseteq Q$ is a set of **final states**. An n-tape FSA accepts an n-tuple of strings if and only if starting from the initial state q_0, it can scan all the symbols on every tape $i, 1 \leq i \leq n$, and end up in a final state $q \in F$.

An **n-tape finite-state transducer** is a 6-tuple $M = (Q, \Sigma, \delta, q_0, F, d)$, where Q, Σ, δ, q_0 and F are like before and $d, 1 \leq d < n$, is the number of **domain tapes**. The number of **range tapes** is simply $n - d$.

Let $A = (Q_1, \Sigma_1, \delta_1, q_1, F_1, d_1)$ and $B = (Q_2, \Sigma_2, \delta_2, q_2, F_2, d_2)$ be two multi-tape transducers over n_1 and n_2 tapes, respectively. Further, let s_i denote the symbol on the ith tape. There is a composition of A and B, denoted by C, if and only if

$$d_2 = n_1 - d_1$$

with

$$C = (Q_1 \times Q_2, \Sigma_1 \cup \Sigma_2, \delta, [q_1, q_2], F_1 \times F_2, d_1)$$

where for all $p_1 \in Q_1$ and $p_2 \in Q_2$,

$$\begin{aligned}\delta([p_1, p_2], s_1 : \cdots : s_{d_1} : s'_{d_2+1} : \cdots : s'_{n_2}) = \\ [\delta_1(p_1, s_1 : \ldots : s_{d_1} : s_{d_1+1} : \ldots : s_{n_1}), \\ \delta_2(p_2, s'_1 : \ldots : s'_{d_2} : s'_{d_2+1} : \ldots : s'_{n_2})]\end{aligned}$$

if and only if

$$s_{d_1+1} = s'_1, \ldots, s_{n_1} = s'_{d_2}$$

The resulting machine is an k-tape machine, where $k = d_1 - d_2 + n_2$.

Implementational Note

We found that it is best not to indicate d, the number of domain tapes, in the data structure representing the automata, but to have it as an argument to the composition function. This enables the user to change the value of d per operation if the need arises.

4.2 A Mixed Grammar

Now we illustrate the advantage of having a linearized lexical form by developing a mixed grammar.

We make use of two grammars for the data presented above. G_1 for templatic nonlinear problems and G_2 for linear issues. For the current data, our G_1 would be similar to the rules in Grammar 1.

G_2 takes as input the output of G_1, i.e., the linearized lexical form such as Syriac */katab/, */waladakatab/, etc. Since R4-R7 in Grammar 2 represent the one phonological phenomenon, viz., the deletion of a short vowel in an open syllable, they can be combined into one rules:

	*	–	a	–	CV	⇔
R8	*	–		–	*	

where C is a consonant and V is a vowel

[6] For regular relations, see (Kaplan and Kay, 1994).

Grammar 3 Grammar for Spirantization, case for [b] → [v]

R9	V	–	b	–	*	⇔	
	*	–	v	–	*		
R10	V	–	⟨c,b,ε⟩	–	*	⇔	
	*	–	v	–	*		
R11	⟨v,ε,V⟩	–	⟨c,b,ε⟩	–	*	⇔	
	*	–	v	–	*		

where V is a vowel

An identity rule (similar to R3 is also required). Applying R8 and the identity rule on the input of G_2 is illustrated below:

w	a	l	a	d	a	k	a	t	a	b	*Linearized Lex Form*
3	3	3	8	3	3	3	8	3	3	3	
w	a	l		d	a	k		t	a	b	*Surface*

Recall that the rule applies right-to-left.

It might not be clear from this example how advantageous is this solution. After all, only three rules were saved. However, note that almost all of the rules in a real grammar do not belong to the templatic morphology domain, but to the linear phonological/orthographic domain. Consider the case of Syriac spirantization mentioned above, viz.,

[- plosive] → [+ fricative] / V ___

Each of the six Syriac plosives requires a set of rules of the form in Grammar 3: R9 applies when the center and context belong to prefixes and suffixes, R10 applies when the center belongs to the stem and the context belongs to a prefix, and R11 applies when the center and context belong to the stem. (Since Syriac stems invariably end in consonants, there is no rule for the case when the center belongs to a suffix and the right context to the stem in this case.) To cover all six plosives, 18 rules are required. If, however, the rules are to apply on the linearized lexical form, each plosive requires only one rule similar to R9 (a total of six rules).

5 Conclusion

Using a linearized form provides a pragmatic solution to the problems discussed above. While the templatic morphology issues are resolved using a multi-tape grammar, the linear-in-nature phonological/graphemic issues are dealt with using a two-tape grammar as in any other Western language. As illustrated with the vowel deletion rule above, this makes the task of the grammar writer easier by far. In addition, the size of the intermediate automata is substantially decreased in terms of space complexity.

There is another advantage of this model if used in a multi-lingual Semitic environment system. We noted above how the derivation of /katab/ in Arabic and Syriac is similar. The only difference is that in the latter a vowel deletion rule takes place. It is then possible to generalize the lexical-to-linearized-form module for more than one Semitic language.

At the abstract finite-state level, our solution may have some similarities with the proposal of (Kornai, 1991) which aims at modeling autosegmental phonology by coding nonlinear autosegmental representations as linear strings. Kornai's approach linearizes the lexical nonlinear representation from the outset using a number of coding mechanisms.

References

Goldsmith, J. 1976. *Autosegmental Phonology.* Ph.D. thesis, MIT. Published as *Autosegmental and Metrical Phonology*, Oxford 1990.

Grimley-Evans, E., G. Kiraz, and S. Pulman. 1996. Compiling a partition-based two-level formalism. In *COLING-96: Papers Presented to the 16th International Conference on Computational Linguistics.*

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

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.

Kiraz, G. 1996. ŞEMḤE: A generalised two-level system. In *Proceedings of the 34th Annual Meeting of the Association for Computational Linguistics.*

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.

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.

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.