Morphological Analysis of Arabic dialect Transcriptions

Stephen Taylor*[1]*, Fitchburg State University

We present some tools for morphological analysis of transcriptions of spoken Arabic from Damascus. The tools are based on Buckwalter's widely used ARAMORPH algorithm and database, and should be easily adaptable to other dialects.

1 Motivation and Structure

We are attempting to model one of the spoken Arabic dialects, *Damascene Colloquial Arabic (DCA*.) Damascus is a cosmopolitan city, and spoken Arabic from all over the world, but especially from all over Syria, can be heard there. Nonetheless, there is a distinct set of features which are usually identified as Damascene. Some of these features are identified in Table 1.

During a Fulbright fellowship in 2009, we collected ten hours of recordings, which we are transcribing and annotating. In addition, we have many hours of commercial recordings of soap operas and television shows. Although these are scripted, non-spontaneous, speech, they share acoustic and grammatical features with “real” spoken DCA.

Our data thus comprise recordings, transcriptions and other annotations. We use several tools for manipulating the data, including Praat [2], the python and .NET programming languages, and a number of small programs. Here we describe an interactive morphological analysis program *MPG*, and off-line tools for customizing its database for DCA.

2 Related Work.

Our work is based on Buckwalter's ARAMORPH morphological analysis program[3]. We have ported his code from PERL to Java and Python, and slightly modified his database formats. The discussion of his algorithm in section 4 applies to his code as well as to our ports of it. The algorithm is based on a database or lexicon, which is specific to an Arabic dialect. Buckwalter's database is directed at Modern Standard Arabic (MSA) which is used by modern newspapers and broadcasters throughout the Arab world. Buckwalter's system and its descendants are widely used in machine translation and Arabic text processing, for example Al Ansary *et al.*[1], although there are many other approaches, currently less widely adopted, including that of Habash and Rambow [7], which I mention below.

Because ours is a system for analysis of dialect text, we need to establish some standard for orthography. Education in the Arab world is conducted in either MSA or Classical Arabic (CA); there are no widely accepted standards for writing or spelling except for these two; and most Arabs conflate them. Both Zawaydeh *et al.*[17] and Buckwalter and Maamouri[4] discuss transcribing colloquial Arabic, as do we. Zawaydeh emphasizes a single transcription for each word, with a prejudice toward an MSA spelling. However, that prejudice is tempered by the fact that foreign and dialect words do not have standard MSA spellings. Buckwalter and Maamouri are even stronger about a preference for MSA spelling, and provide a limited list of exception words. They do not place the same emphasis on a single transcription, but the emphasis on MSA spelling has the same result. In both cases, there are guidelines for non-MSA affixes. Some reflections on these guidelines at the end of the project are

1 staylor@fitchburgstate.edu http://falcon.fsc.edu/~staylor

found in Graff *et al.*[6] We discuss our transcription guidelines in section 3.

In section 5 we describe the grammar used by our tool for generating new prefix and suffix entries for Buckwalter's morphological database. Habash and Rambow[7] describe similar changes in their overview of adding changes to their *Magead* morphological analyzer to handle Levantine Arabic.

The problem of adding new stems or new headwords to a dictionary is central to database-oriented approaches like the Buckwalter algorithm. The database is under steady development at the Linguistic Data Consortium. I am not aware of any publications describing their process for insertion of new lexemes, but Graff *et al.*[6] report on the process used for building a dictionary for Iraqi Arabic at the LDC, apparently from scratch. Maamouri *et al.*[10] discuss several approaches for Levantine Arabic dictionary building. Shaalan[12] reports on a project of adding Egyptian Colloquial Arabic stems to the Buckwalter lexicon both by hand-edits and by rule-based transformations of MSA stems already in the database. Our approach in section 6 slightly automates the process of adding new verb stems, but it does nothing to automate the process of finding them.

Sections 7 and 8 present our tags and the tagging tool *MPG* we wrote to record them in the transcription file. We do serial, in context tagging, that is, each common wordform will be tagged each time it occurs in the text. This contrasts with the approach in Graff *et al.*[6] in evaluating all identical wordforms at once.

3 The transcriptions.

Our transcriptions are in Arabic script, and look almost like ordinary written Arabic. This is a deliberate decision based on two different goals: They are more accessible to native speakers than a romanized transcription like IPA, and they can be related more easily to existing glossaries, in particular the well-known *ARAMORPH database* [3]. We type our transcriptions using Praat, and the transcriptions are synchronized with the recordings using *Praat textGrid format* (Boer*sma).* Here is an example:

يلي بيعرف بيعرف يلي ما بيعرف بيقول كف عدس

[This is a common saying, in romanized transcription: */yilly byعrif byعrif yilly maa byعrif by'ul kef عdas/ (He who knows, knows, he who doesn't know, says "A handful of beans.")* It clearly has an explanatory story, since it doesn't make much sense by itself. In an effort to obtain spontaneous speech, I asked people to explain common sayings in their own words.] The transcription is intended to emphasize both continuity with CA and differences from it. Here are a few of its features:

- Not visible in this excerpt, is the fact that the transcription is divided into short segments, marked with timing information which synchronizes it with the recording; also not visible is the existence of additional *tiers* of annotation. In addition to the transcription, there are a *words* tier, a *phones* tier, and a *tagging* tier, each marked with timing information.
- It lacks punctuation. Praat accepts Arabic input, but it treats punctuation, including Arabic commas, as interposed English, so it moves the commas, periods, or quotation marks to the right-hand end of the word. Rather than fix this in Praat, we omit punctuation, or surround it with spaces.
- It lacks vowels. This is normal in written Arabic, and makes the text look more familiar to native speakers, as well as easier to type. Since the vowel sounds are available in the recordings, the information is not lost. As will be seen, the morphological analysis includes vowels, but these might be in error.

- The world */yily/* يلي (that) is a DCA relative pronoun. It is not an MSA or CA word. (There is a perfectly good CA word with a similar meaning, but different pronunciation and spelling.)
- */by ٤rif/* بيعرف (he knows) and /by'ul/ بيقول (he says) have a /*b*/ ب (present indicative prefix.) This prefix, which does not occur in CA, is one of the grammatical differences between CA and DCA.
- The negation is expressed with /maa/ ما instead of /laa/ لا as it might be in Classical Arabic.
- The glottal stop in /by'ul/ بيقول (he says) is spelled with a *qaaf* ق, reflecting the classical spelling of the word, which makes it more familiar to the reader (and easier for a program to lookup in a dictionary) even though the reader will have seldom seen the word with a /b/ ب prefix. We tend to spell words following classical spelling when the classical consonant is */qaaf/* ق, */thaa/* ث, */dhaal/* ذ, in spite of the fact that each of these consonants has two or more common colloquial pronunciations, and thus the orthography does not follow the phonetics. As is the case with vowels, the information is available in the recordings. In a few common words, we do modify the spelling. In general, our transcriptions reflect the history of the word and not its pronunciation.

The relevant features are summarized in Table 1:

Table 1. *Some features of DCA as contrasted with CA*

Feature	**Examples**	**In Transcription?**
Accent: Multiple onset consonants	*Relax!* /striyH/ستريح *Much* /ktiyr/ كتير *Three* /tlaate/ تلاتة	Invisible, if vowels not marked. Appears in phones tier.
Accent: ق → ء	*Heart* → /'alb /qalb/ / قلب → ألب	No.
Accent: ث → ت or س	كثير كتير ثلاثة → تلاتة	For common words. Also, sometimes as an error.
Accent: ذ → د or ز	هذا → هادا ماذا مازا	For common words. Also, sometimes as an error.
Accent: lengthened final vowel	Feminine nouns ending in ة	No
Accent: lengthened final vowel	Masculine possessive pronoun ه pronounced as و following a consonant.	Usually not.
Grammar?: هادا, هاي, عاى merges with following article	هادا → هالكتاب الكتاب	Inconsistent
Grammar: Verb inflections	*I write* /ktub/ (versus CA /'aktub/) *They write* /yaktubuw/ (versus CA /yaktubuwn/)	Yes.
Grammar: uninflected راح or ح as future tense marker	راح روح *I will go*	Always.

Grammar: ب- as a present indicative marker	بـروح *I go*	Always.
Grammar: object pronouns	هم -> هـون كم -> كـون	Yes.
Vocabulary:	شـوء what لـيش *why* ويـن *where* مـنين *whence*	Yes.

The transcription guidelines are similar, but not identical, to Buckwalter and Maamouri's guidelines for transcribing Levantine Arabic [4] for the Call Home project. In both cases, it is a goal to make the text readable without distorting the underlying vocalization, and in both cases, dropping some information in the transcript is justified because there are other levels of annotation. In fact, we have as yet produced very little of this additional annotation, but we project that we will eventually fill in the following Praat *tiers*:

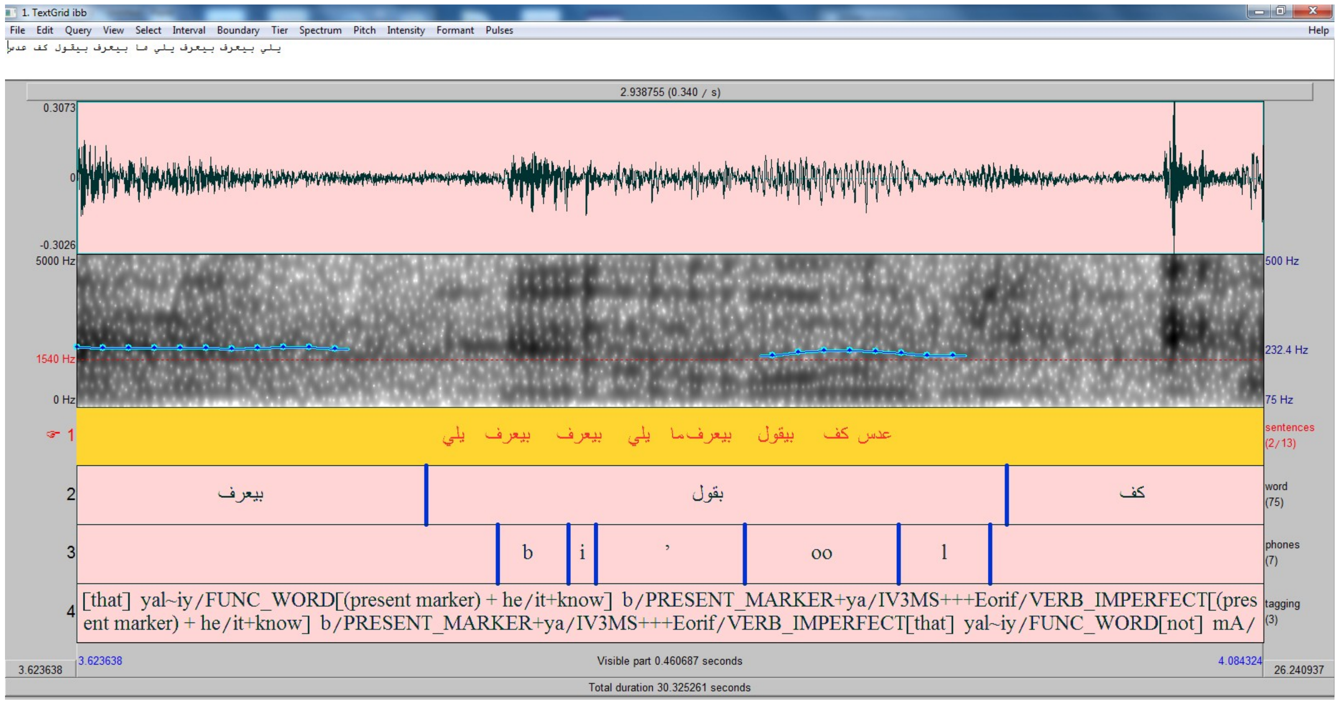
1) The *sentence* tier is the transcription we have just described. Many of the segments are in fact not complete sentences, but most are multi-word phrases.
2) The *word* tier marks word boundaries. Presumably each sentence segment corresponds to one or more word segments; only a few of our recordings have this tier populated. It seems plausible that the word tier could be computed, possibly with hand-edited corrections. In this scenario, speech recognition technology would align the indicated word from the sentence-level transcription with the recordings. The transcription for the word tier segments is identical to the transcription for the sentence tier, except that each segment contains only one word.
3) The *phones* tier marks phonemes. The plan is for the phoneme tier to use a non-Arabic alphabet. This allows for a wider range of vowels than CA provides and unambiguous consonant sounds. Our current phoneme annotation is based on Stowasser's[14] transliteration scheme for DCA, but we still have so little phoneme annotation that we are not committed to a particular notation. Again, it seems plausible that the phonemes could be computed from the word boundaries and the recordings, and that, in fact, both calculations should be carried out together. A somewhat similar computation is described in (Vergyri04.)
4) The *tagging* tier contains a morphological analysis, described in much greater detail below. Briefly, it provides morphological tags and an English gloss for each morpheme. Since the breakup of a wordform into morphemes is in general ambiguous, this tier resolves which of the possible meanings for a wordform we understand to be intended in context.

The following Praat screenshot illustrates several of these features, including

1) The editing window is the band across the top. It shows the contents of the transcript segment, tier 1, which is currently selected. The editing window is the only one in which the Arabic text appears correctly in the current version of Praat.
2) The recording is displayed. The time flows from left-to-right.
3) Various computed properties of the recorded sound are displayed, including a spectrogram and a pitch line.
4) The transcript segment is currently selected, thus the words from the transcript segment appear in the edit panel. The words of the transcript segment are rearranged, so that they follow the flow of time on the recording.

5) The word tier shows individual words, with boundaries between marked between word segments. Since they are aligned with the recording, they also read from left-to-right, although each individual word is correctly displayed right-to-left.
6) The phone tier is still largely unpopulated, because deciding on which vowels are intended and where each phoneme begins and ends is a lot of work we haven't done yet. Here phonemes are marked for */by'ul/* بيقول (he says)
7) The tagging tier shows morphological tags for the transcript segment.

Figure 1. *Praat Screen Shot*



يلي بيعرف بيعرف يلي ما بيعرف بيقول كف عدس

1 يلي بيعرف بيعرف يلي بيعرف‌ما بيقول كف عدس

2 بيعرف | بقول | كف

3 b | i | ' | oo | l

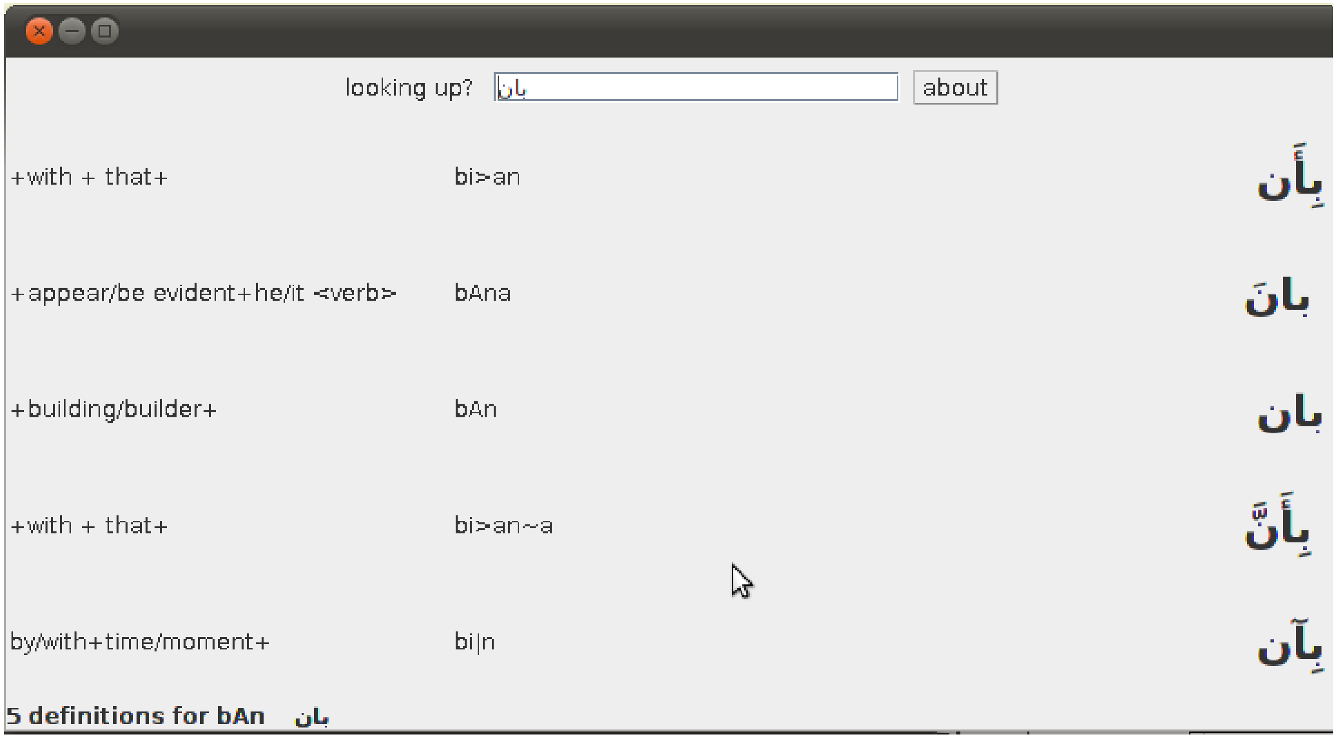
4 [that] yal~iy/FUNC_WORD[(present marker) + he/it+know] b/PRESENT_MARKER+ya/IV3MS+++Eorif/VERB_IMPERFECT[(present marker) + he/it+know] b/PRESENT_MARKER+ya/IV3MS+++Eorif/VERB_IMPERFECT[that] yal~iy/FUNC_WORD[not] mA/

4 Buckwalter's Morphological Analyzer

Tim Buckwalter's ARAMORPH program/database[3] has been incorporated into many open-source Arabic tools. It is a program for suggesting morphological analyses for media Arabic, primarily news articles. It contains a database of prefixes, stems, and suffixes, and their meanings, which permits it to be used as an electronic dictionary. This screenshot, which shows a port of ARAMORPH to Java, illustrates the idea. This particular port isn't showing the morphological breakup of the wordform, or the part-of-speech tagging.

The word being looked up is typed at the top. The three columns in the panel show an English gloss, built up from the glosses of the prefix, the stem, and the suffix, the internal representation of this word using the Buckwalter transliteration in which the database is stored, and an Arabic rendering of the vocalized word.

Figure 2. *A sample run of the ARAMORPH algorithm.*



looking up? بان about

+with + that+ bi>an بِأَن

+appear/be evident+he/it <verb> bAna بانَ

+building/builder+ bAn بان

+with + that+ bi>an~a بِأَنَّ

by/with+time/moment+ bi|n بِآن

5 definitions for bAn بان

The ARAMORPH algorithm is based on trying every possible split of the word into prefix, stem, and suffix, and a database, or lexicon, containing all the allowable instances of each. The prefix and the stem can be empty, but there must always be at least one letter in the stem.

The following table illustrates the steps in the algorithm.

Table 2. *Steps in the Buckwalter algorithm*

Suffix-key	Stem-key	Prefix-key		(Possible) Words
(none)	بـا ن	(none)		بـا نَ بـا ن
ن	بـا	(none)		No such stem
ا ن	ب	(none)		Stem and Suffix incompatible
(none)	ا ن	ب		بِـأَن بِـأَنَّ بِـآن
ن	ا	ب		Prefix and stem inconsistent
(none)	ن	بـا		No such prefix

As each possible split of the letters is generated, we check to see whether each of the prefix, stem, and suffix exists in the database, and whether its category is compatible with the others.

This simple example doesn't illustrate it, but a prefix may be built up of smaller prefixes, for example the prefix /wabil-/وبال (and in the) is made up of the conjunction /wa/و (and), the preposition /bi/ب (in), and the definite article, /al/ال (the). Suffixes may similarly be compounds, but stems are usually not. The prefix and suffix databases contain complete affixes. For our example, there would be database entries for /wa-/, /bi-/, /al-/, /wal-/, /bil-/, /wabi-/, and /wabil-/

Here are a few sample rows from the lexicon database, chosen to match the example:

Prefixes:

```
,,Pref-0,,,
b,bi,NPref-Bi,by/with,bi/PREP+,
```

Stems:

```
A,>a,FW,A/1st,>a/ABBREV,>a_1
An,>an,FW-Wa,to,>an/FUNC_WORD,>an_1
An,>an~a,FW-Wa-n~a,that,>an~a/FUNC_WORD,>an~a_1
An,<in~a,FW-Wa,that/indeed,<in~a/FUNC_WORD,    <in~a_1
An,>an~,NapAt,moan/complaint,>an~/NOUN,>an~ap_1
An,>an,PV_ttAw,mature/approach,>an/VERB_PERFECT,    >anaY-i_1
An,>on,IV_0hwnyn_no-Pref-A,mature/approach,>on/VERB_IMPERFECT,    >anaY-i_1
An,|n,PV_V,arrive/approach,|n/VERB_PERFECT,|n-ui_1
bAn,bAn,NK,building/builder,bAn/NOUN,bAniy_1
bAn,bAn,PV_V_intr,appear/be      evident,bAn/VERB_PERFECT,bAn-i_1
bAn,bAn,IV_V_Pass_yu,be  explained,bAn/VERB_IMPERFECT,>abAn_1
n,n,FW,N/14th,n/ABBREV,n_1
```

Suffixes:

```
,,Suff-0,,,
,o,CVSuff-o,you <verb>,+o/CVSUFF_SUBJ:2MS,
,a,PVSuff-a,he/it <verb>,+a/PVSUFF_SUBJ:3MS,
An,Ani,IVSuff-An,[du.],   +Ani/IVSUFF_SUBJ:D_MOOD:I,
An,Ani,NSuff-An,two,+Ani/NSUFF_MASC_DU_NOM,
n,ni,NSuff-|-ni,two,+Ani/NSUFF_MASC_DU_NOM,
n,ona,PVSuff-n,they [fem.pl.] <verb>, +na/PVSUFF_SUBJ:3FP,
n,~un~a,PVSuff-~t,you [fem.pl.] <verb>, +tun~a/PVSUFF_SUBJ:2FP,
n,ona,IVSuff-n,[fem.pl.],+na/IVSUFF_SUBJ:FP,
n,ni,IVSuff-|n,[du.],+Ani/IVSUFF_SUBJ:D_MOOD:I,
```

The database also contains compatibility tables for Prefixes and Stems**,** Prefixes and Suffixes**,** and Stems and Suffixes..

Each of the Prefix, Stem, and Suffix entries is in a similar format. I have shown the fields separated by commas. The first field is the key, the unvowelled letters of the morpheme. For example, we see that the key /An/ addresses six different stem morphemes and two suffix morphemes. The second field is the vocalized spelling of the morpheme. All of the stem morphemes /An/ have different vocalizations, but the two suffixes have the same vowels. The third field is the *category.* The category is central concept of the ARAMORPH algorithm, and is used in the compatibility tables to determine whether a given prefix, stem, and suffix are compatible, and can form a word. For example, the first /An/ suffix is category *IVSuff-An,* which is compatible with imperfect verb stems and first and second person imperfect verb prefixes. The second /An/ suffix is category *Nsuff-An,* which is compatible with most noun stems. The fourth field is an English gloss, the fifth field is a part-of-speech tag, and the sixth field is a stem name.

5 The Morphological Database – Customizing Prefixes and Suffixes

Editing the database by hand is perfectly feasible, but repetitive and error-prone. Instead of doing so, we wrote a simple grammar, which can be used to build the prefix and suffix tables. The grammar lets us group similar edits, and even though the rules are wordy, the grammar file is about a quarter the size of the generated database table. We call the grammar format *y2*, because it is our third iteration of the design. The first attempt was the *.y* format used by *yacc,* the compiler-compiler tool. *Y2* is much less powerful than *.y*, but it contains the features we need to create the affix lexicons. The grammar is compiled by a python program named `ready2.py.`

Here are some fragments of our grammar, `prefixes.y2`, for generating the CA prefix table. It generates almost the same entries as Buckwalter's original lexicon, except that some of the English glosses are more consistent (and correspondingly less helpful:)

```
export NPref-Li IVPref-AnA-| IVPref-AntmA-tu IVPref-nHn-linu
...
     %token alif_wasla='{'
     %token hamza_alone='''
...
/*  noun prefixes */

conj    :      waw fatha      {wa/CONJ}  [and]
     |      faa fatha      {fa/CONJ}  [and/so]

prep_bi :      baa kasra      {bi/PREP}      [with/by]
     |      kaaf fatha      {ka/PREP}      [like/such as]

prep_li :      lam kasra      {li/PREP}      [for/to]

la      :      lam fatha  {la/EMPHATIC_PARTICLE} [indeed/truly]

/* could be alif_wasla, but spelling is rare. */
al      :      alif lam       {Al/DET}       [the]

Pref-Wa :      conj

NPref-Bi:      prep_bi
     |      conj prep_bi
```

The *export* lines indicate which description symbols are exported to the database. In this case, the description symbols will be entries in the category fields of the prefix part of the database. Since all of the original categories in the Buckwalter database began with capital letters, we adopted the convention of beginning non-exported symbols with lower-case characters.

The *token* lines define the alphabet symbols, here the Buckwalter transliteration used in the database. Once they are defined, the alphabet symbols are used just like the description symbols, but the token lines do not allow for giving alphabet symbols a gloss or a part-of-speech. [The Buckwalter transliteration has a one-to-one relationship to the Unicode Arabic characters, but is useful for software development because most text editors still do not handle mixed Latin and Arabic characters gracefully.]

Each descriptive symbol, like *conj, la,* or *NPref-Bi,* is defined exactly once, by naming the symbol and following it with a /:/ (colon).

The symbol name is followed by a *rule,* which consists of a

- *right-hand-side,* which is a (possibly empty) list of symbols, either alphabet symbols or descriptive symbols;
- optionally a part-of-speech labeling, which, if present, appears in *{curly brackets}* and
- optionally an English gloss, which, if present, appears in *[square brackets.]*

Some of the descriptive symbols have single-line definitions, but many are followed by additional lines giving alternative definitions. An alternative definition begins with a /|/ (vertical bar), which is followed by a rule. Every descriptive symbol definition includes at least one rule, but there may be many rules, all but the first being alternative definitions.

If the part-of-speech and gloss do not appear in the rule, they are inherited from the constituents of the right-hand side of the rule.

Thus the rule:

NPref-Bi: **prep_bi**

generates two lines in the database, corresponding to the two lines of the definition of *prep_bi*

Table 3: *Database columns*

Key	Vocalized	Category	English Gloss	POS label
b	bi	NPref-Bi	by/with	bi/PREP+
k	ka	NPref-Bi	like/such as	ka/PREP+

(These lines actually appear as:

b,bi,NPref-Bi,by/with,bi/PREP+,
k,ka,NPref-Bi,like/such as,ka/PREP+,

The empty sixth field is used for stem names (empty for these prefixes) which are not used by the analysis algorithm, but are useful for comparing database rows with printed dictionaries.)

The alternative rule

| **conj prep_bi**

generates four more, combining the two lines of *conj* with the two lines of *prep_bi*:

Table 4: *Database columns*

Key	Vocalized	Category	English Gloss	POS label
wb	wabi	NPref-Bi	and + by/with	wa/CONJ+bi/PREP+
fb	fabi	NPref-Bi	and/so + by/with	fa/CONJ+bi/PREP+
wk	waka	NPref-Bi	and + like/such as	wa/CONJ+ka/PREP+
fk	faka	NPref-Bi	and/so + like/such as	fa/CONJ+ka/PREP+

Here are a few of the added rules in our file `brefixes.y2`, which is used to generate the DCA prefix table:

```
/* present marker phrase */
pm      : baa    {b/PRESENT_MARKER} [(present marker)]

p_m     : pm
        | conj pm

/* present marker phrase for 1P */
pn      : mim   {m/PRESENT_MARKER} [(present marker)]

p_n     : pn
        | conj pn

 /* imperfect conjugations with present (indicative) marker */
IVPref-PM-hw-ya       : p_m ya          {^3MS}          [he/it]

IVPref-PM-hw-yu       : p_m yu          {^3MS}          [he/it]

IVPref-PM-hm-ya       : p_m ya          {^3P}           [they]

IVPref-PM-hm-yu       : p_m yu          {^3P}           [they]
```

The only new notation here is the POS label **{^3MS}** which indicates that the letters **3MS** should be added to the POS label generated from **p_m** and **ya (**or **yu**.) The caret shows where to place the generated label, which in this instance will result in **b/PRESENT_MARKER+ya/IV3MS.** IV3MS stands for Imperfect Verb 3rd person Masculine Singular. The POS label on the **IVPref-PM-hm-ya** and **IVPref-PM-hm-yu** symbols will be in **b/PRESENT_MARKER+ya/IV3P.** DCA doesn't have a gender distinction in the plural, hence the **3P** instead of **3MP**. The names of these symbols are similar to the corresponding symbols for the categories already in the database. This helps us in specifying which suffixes can go with them. The prefixes themselves are exactly the same, but they have a different gloss and a different category, and the compatibility tables will only allow the plural prefix with an appropriate plural suffux.

Including using the caret notation, there are several possibilities for generating a gloss or POS label:

1) No label is given in the rule, and the new label is constructed by inserting a /+/ (plus sign) between the labels of the non-terminals of the rule's right-hand-side.
2) A label without a caret is given in the rule, and it is the label for this rule. The labels of the constituents are ignored.
3) A label is given which includes a caret: a string is constructed from the constituents of the right-hand-side as in (1) which replaces the caret in the label for this rule.

Sample database lines for **IVPref-PM-hw-ya** and **IVPref-PM-hm-ya:**

Table 5: *Database columns*

Key	Vocalized	Category	English Gloss	POS label
by	bya	IVPref-PM-hw-ya	(present marker) + he/it	b/PRESENT_MARKER+ya/IV3MS+
by	bya	IVPref-PM-hmA-ya	(present marker) +they	b/PRESENT_MARKER+ya/IV3P+

Note the double consonants in the onset of the vocalized prefix.

Since we introduce new categories, it is necessary to add them to the compatibility tables. We hand-edited these lines into files, which we then append to the existing tables.

This example happens to highlight a significant problem with our prefix scheme: the vowels are wrong. The correct vowel for the /y-/ي prefix in DCA is a /schwa/ ə, for almost all cases in which there is a vowel. The sound of a schwa is close to /damma/ُ so revising the compatibility tables to always use an IV…yu prefix sounds like a possible fix, without much effort. However, the picture is more complicated[5]. If the first letter of the imperfect stem is a consonant, the vowel appears between the /b/ب (present marker) and the person-prefix, so that the prefix might become /bət-/بْت. For the semi-vowel /y/ي prefix in the example, if the verb begins with a single consonant, the prefix is typically replaced with a /i/(kasra) so the example prefix would have been /bi-/بِ instead of /byu-/بيُ. Any or all of these changes fits easily in the range of what can be handled by the Buckwalter algorithm, but implementing *any* of them (but especially dropping the /y/ي(yaa)) would break MSA compatibility in the transcription. The ultimate solution might be to implement a separate phonetic column in the database entries. This would also address the neglected problem of DCA consonants with two pronunciations.

(To recap the problem, MSA orthography is not phonetic, but once the vocalization is determined, the pronunciation is determined, because the consonant sounds are constant, and the stress can be determined easily once the syllables are marked. Similarly, once we have determined which word we are representing in DCA, the pronunciation is available, perhaps by another table lookup. The word /soora/ ثورة (revolution) and the word /tlaate/ثلاثة (three) are never pronounced /toora/ or /slaase/. Once we have determined the word we are looking at, we can find the pronunciation. The stem database seems like a perfectly good place to store this information, but it is not there now. Unlike MSA, DCA needs more than the vowels to determine the pronunciation.)

6 The Morphological Database – Adding new Verb Stems

The major problem with the database-dependent ARAMORPH algorithm is the open morphological classes: nouns, adjectives, and verbs. Although the database contains 82,083 variations on 38,573 stems, it is slanted toward news articles, and outside its primary domain almost every sentence contains one or more words not in the database. In particular, our transcripts contain DCA words which are either new to the database or have different common meanings. Our tagging tool therefore allows manual insertion of tags; but a better long-term solution is to extend the database by adding new stems. We add new particles, nouns and adjectives manually, but we've written a program for adding new verb stems.

The reason that verbs are particularly interesting has to do with how verb stems appear in the database. The stems are retrieved from the database without modification. Since the database entries show the vowels for the stem, and the internal vowelling of imperfect and perfect stems differ, there must be at least two entries for each verb, one for the imperfect stem, and one for the perfect stem. For example, the two stems */-ktub-/* كتُب and */katab/* كَتَب are needed for the verb (to write.) But many verbs are more complicated than that. For example, the third person perfect stem of the verb */saawaa/* ساوى (do/make) is the citation form, but if there is an object pronoun, the verb-stem must be */saawaa-/* ساوا (do/make) because */aa/* ى (alif maqsura) can only appear at the end of a word. The perfect stem for the consonant stem-suffixes is */saawiy-/* ساوي (do/make) e.g. */saawiit/* ساويت (I made), and for vowel suffixes */saaw-/* ساو (do/make), e.g., */saawoo/* ساووا (they made.) Similarly, there are multiple entries for the imperfect verb stem. Each entry has a category appropriate to its peculiarities, so that the compatibility tables of the database permit it to combine in only morphologically correct ways. The genius of the Buckwalter algorithm is that these rule-driven, but complex, decisions are precompiled into the database; rare exceptions to the rules can be hand-edited.

Our contribution is to provide a grouping of verbs which simplifies the rules. Following Hussein Maxos, we divide most verbs into ten major groups, based on similarity of their inflections. Each major group has 2-4 subgroups.

In our grouping, groups 1-4 all correspond to the Form I classification, that is, they all have the form /f عl/فعل. Verbs in group 1 all have the form /f عal/فـعَـل and verbs in group 2 all have the form /f عil/فعِل ; verbs in groups 1A and 2A are strong, and verbs in groups 1B and 2B are final-weak verbs. Our group 3 corresponds to hollow verbs (/faal/فال , /fuul/فـول , or /fiil/فـيـل), group 4 to geminates (/f عll/فعلّ.) Our group 5 is Form II (/f ععl/فعّل), our group 6 is Form V (/taf عl/تَفعل), Group 7 is approximately Form III (/faa عl/فَاعل), group 8 is Form VIII (/Ifti عal/أفتِعَل), group 9 is Form X(/Istaf عl/آستَبعل), and group 10 is form IV(/Af عal/أبعَل). Groups 1, 2, 5, 6, 7, 8, and 9 are subdivided for final weak verbs. (Forms VI, VII, IX were not mentioned, but verbs in form VII /Inf عl/إنفعل) are the passives of verbs in groups 1, 2, 3, 4, and can be dealt with there, although we are not currently doing so. Verbs in form VI (/tafaa عl/تفاعل) are the passives of verbs in group 7. We have not yet dealt with verbs in form IX (/If عll/إفعلّ) or forms XI-XV..

This division seems to result in fairly concise rules for transforming stems.

Input to our database insertion tool consists of a text file. The text file is organized by sections of verbs in the same verb group, separated by *#group:* headers. Each line within a section contains a partially vocalized verb inflected for the third person singular imperfect and a gloss. These three items of information are sufficient to derive the database entries for the verb. In the simplest case, there is only one perfect stem and one imperfect stem for each verb, but in more complicated cases, e.g. final-weak verbs, there may be six or more. (The most complicated verb already in the database, */laisa/*لَيسَ (is not) has 41 entries.)

It is well-known that almost all Arabic verbs can be completely inflected by applying a large collection of simple rules, so in that respect our tool does not break new ground. However, our grouping scheme seems to result in very simple code for each group or group subdivision.

7 Morphological tags

While we were transcribing the data, we built an MS Access database containing as much as we could manage about each new word. As we morphologically tag the transcriptions, we incorporate the information into the morphological database, which becomes increasingly useful as a DCA to English dictionary.

The process of morphological tagging is a test of the coverage of our models and of the lexica based on our data. The tags themselves are useful for frequency counts of stems, for back-off frequencies of n-grams (see Wang[16], for example) and for other grammatical models.

Our tags incorporate an English gloss, a Buckwalter transliteration with vowels, and part of speech tags.

1) Since the majority of stems come from the MSA-based Buckwalter lexicon, the vowels may not reflect DCA usage, but we present them (though in transliteration) as useful for a native speaker to compare our disambiguated word choice with her own judgment.
2) The English gloss is at best a loose approximation of the meaning, but it still speeds the tagging

process.

3) We do not record some of the information returned by the analysis process. For example, the stem labels could be useful for frequency counts; recomputing the stem label is possible by repeating the analysis and comparing our tags with the possible outputs.

The following table illustrates the tags.

Table 6: Some sample tags

English Gloss	Transliteration and POS tagging	Arabic
That	yal~iy/FUNC_WORD	يلي
(present marker) + he/it+know	b/PRESENT_MARKER+ya/IV3MS++ +Eorif/VERB_IMPERFECT	بيعرف
(present marker) + he/it+know	b/PRESENT_MARKER+ya/IV3MS++ +Eorif/VERB_IMPERFECT	بيعرف
That	yal~iy/FUNC_WORD	يلي
not	mA/NEG_PART	ما

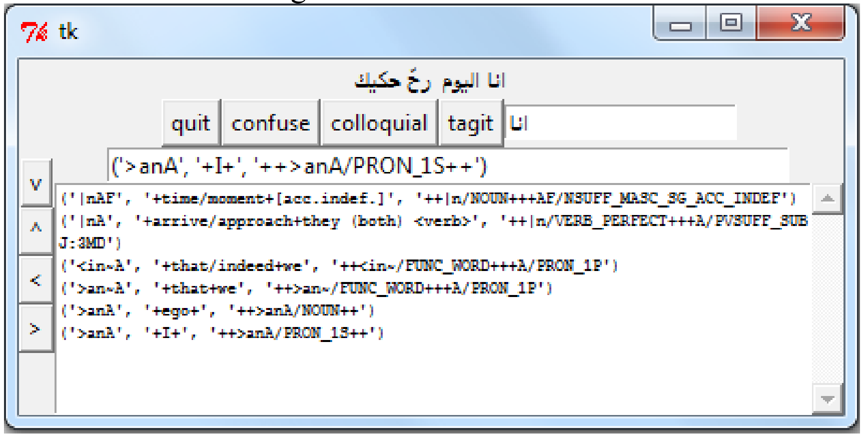
8 The tagging tool, *MPG*

One could manually tag the text, but since we hope to apply simple computer programs to process the tags, it is desirable for them to be as consistent as possible. Using a computer program to do tagging eases this process of producing consistent tags. We have written several tagging programs, all of which use the same morphological analysis module we have just described. Our current tagging program is called *MPG* (for *M*orphological, *P*OS, and *G*loss tags.) For each word, it offers a number of possible analyses. The user chooses or edits one of the possible tags, or creates one from scratch. So long as the morphological analyzer can handle a word, the process is quick, with only a few keystrokes required to choose and enter a tag. As soon as the morphological analysis fails (usually because of a stem missing from the stem lexicon) the process slows dramatically.

In order to make use of as much as possible of the the existing Buckwalter lexicon, one of the options while tagging is to substitute letters in the sample word. As shown in Table 1, there are several substitutions which commonly occur in pronouncing CA words in DCA, and sometimes these substitutions of pronunciation appear in our transcripts. If the word does not appear in the lexicon at all, the tag may be entered by hand, but it is faster to chose among automatically generated tags, by trying alternate spellings. There are two ways to do this, either by editing the word presented to the morphological analyzer, or by choosing a *Confuse* option, which presents computed tags for all the common substitutions for the given word. This spelling correction could in principle deal with incorrect word segmentation also; we have a stand-alone program which does so. But it generates too many alternatives to comfortably scroll through, and we chose not to incorporate this feature into our tagging tool.

The tagging tool, MPG, is an interactive python program. The morphological analyzer it employs, *bw.py*, is implemented as a python module (we also have Visual Basic and Java implementations.) The following screenshot shows the tagging tool in operation.

Figure 3: MPG screenshot



انا اليوم رحْ حكيك

انا

('>anA', '+I+', '++>anA/PRON_1S++')

```
('|nAF', '+time/moment+[acc.indef.]', '++|n/NOUN+++AF/NSUFF_MASC_SG_ACC_INDEF')
('|nA', '+arrive/approach+they (both) <verb>', '++|n/VERB_PERFECT+++A/PVSUFF_SUB
J:3MD')
('<in~A', '+that/indeed+we', '++<in~/FUNC_WORD+++A/PRON_1P')
('>an~A', '+that+we', '++>an~/FUNC_WORD+++A/PRON_1P')
('>anA', '+ego+', '++>anA/NOUN++')
('>anA', '+I+', '++>anA/PRON_1S++')
```

The transcription segment from the Praat textgrid sentence tier appears in the top line. (The sentence tier of the file is not changed by tagging.) The arrow buttons on the left permit navigation between phrases and among words in a phrase. The current word appears in a box toward the northeast of the frame. Beneath it is candidate tag information, either chosen from the options in the scroll box, or hand-edited. The tagger operates in either DCA or CA mode; here it is in CA mode, but can be switched with the *colloquial/classical* button. The analysis algorithm is the same for either mode, but the databases differ. The *confuse* button generates tags for common letter substitutions, and the *quit* button writes all the tags to a *tagging* tier in the textgrid file as it exits, so that the tags appear in parallel with the sound and the transcription.

9 Conclusion and future work

The various components comprise a system, under continuing development, for morphological analysis of colloquial wordforms in transcriptions of recordings. They include:

- The transcription format and guidelines.
- Buckwalter's algorithms and database, ported to new programming environments.
- Tools for rebuilding and extending the prefix and suffix lexica, and source files for the tools.
- A table of common character substitutions, and a simple algorithm for using the table.
- A tagging tool which makes use of all of these.

All of the components are available on the web at

http://falcon.fsc.edu/~staylor/Syrian

Our experience of developing a DCA morphological analyzer relatively quickly convinces us that our transcription guidelines are reasonable. Maintaining MSA orthography permitted us to reuse the Buckwalter stem lexicon with relatively modest changes. We also believe that native speakers take the transcriptions more seriously if they look more like MSA.

Of our tools, the single component we believe most likely to be useful for other dialects is the prefix and suffix grammar. For example, Egyptian Colloquial Arabic has a number of features in common with DCA, and almost all of the DCA grammar would be relevant to ECA, but it has some features which DCA lacks. For example, a morphological analyzer for ECA should handle the negation suffix. (Without trying it we believe) adding rules to our grammar files to do so should be straight-forward.

Our choice of Praat textGrid files to hold our parallel data tiers has worked well for us. We are using

Praat for the acoustic tagging, editing the transcripts and word and phoneme boundaries. Adding the MPG tagging to the same file keeps all the data in one place. Other formats, including XML, would work as well for aggregatin the data, but would need conversion to work with Praat, We have a python module for reading and writing the textgrid format, which could be used by others.

Our python module for carrying out the Buckwalter morphological analysis algorithm can be used in dictionaries and spelling checkers, in addition to our use in a tagger.

Some ideas for future work:

1) We hope to find tools to use the acoustic information we have available.
2) The database could be extended to handle a DCA phonetic representation, or an additional tool, possibly with a lexicon of its own, could be built to do this.
3) Our tool for inserting verb stems could be extended to handle passives.
4) The database can be used for an English-DCA dictionary, but for this use frequency information would greatly increase its utility.

References

[1] ALANSARY, S., NAGI, M., , AND ADLY, N. Toward Analyzing the International Corpus of Arabic (ICA): Progress of Morphological Stage. In *8th International Conference on Language Engineering* (2008).

[2] BOERSMA, P., AND WEENINK, D. Praat: Doing phonetics by computer (version 5.1.05) [computer program], 2009. http://www.fon.hum.uva.nl/praat/ [as of 11/16/2010].

[3] BUCKWALTER, T. Aramorph 1.0 computer program, 2002. LDC catalog no. LDC2002L49.

[4] BUCKWALTER, T., AND MAAMOURI, M. Guidelines for the transcription of Arabic dialects (EARS), 2004. [http://projects.ldc.upenn.edu/EARS/Arabic/Guidelines_Levantine_MSA.htm as of March 2011]

[5] COWELL, M. W. *A Reference Grammar for Syrian Arabic*. Georgetown University Press, 2005.

[6] GRAFF, D., BUCKWALTER, T., JIN, H., AND MAAMOURI, M. Lexicon Development for Varieties of Spoken Colloquial Arabic . In *International Conference on Language Resources and Evaluation* (2006).

[7] HABASH, N., AND RAMBOW, O. Morphophonemic and orthographic rules in a multi- dialectal morphological analyzer and generator for arabic verbs. In *ACL* (2007).

[8] HAJIC, J., SMRZ, O., ZEMANEK, P., PAJAS, P., SNAIDAUF, J., BESKA, E., KRACMAR, J., AND HASSANOVA, K. Prague Arabic Dependency Treebank 1.0, 2004.

[9] HOLES, C. *Modern Arabic Structures, Functions and Varieties*. Georgetown University Press, 2004.

[10] MAAMOURI, M., BIES, A., BUCKWALTER, T., DIAB, M., HABASH, N., RAMBOW, O., AND TABESSI, D. Developing and Using a Pilot Dialectal Arabic Treebank. In *International Conference on Language Resources and Evaluation* (2006).

[11] MAAMOURI, M., GRAFF, D., JIN, H., CIERI, C., AND BUCKWALTER, T. Dialectal Arabic Orthography-based Transcription & CTS Levantine Arabic Collection. In *EARS RT-04 Workshop* (2004).

[12] MAXOS, H. Arabic Verbs Conjugation. In *Arabic for non-natives – Modern Arabic Grammar* (2009), self-published.

[13] SHAALAN, K., ABU BAKR, H. M., AND ZIEDAN, I. Transferring Egyptian Colloquial Dialect into Modern Standard Arabic . In *International Conference on Recent Advances in Natural Language Processing* (2007).

[14] STOWASSER, K., AND ANI, M. *A Dictionary of Syrian Arabic*. Georgetown University Press, 2004.

[15] VERGYRI, D., AND KIRCHHOFF, K. Automatic diacritization of Arabic for acoustic modeling in speech recognition. In *In COLING 2004 Computational Approaches to Arabic Script-based Languages* (2004).

[16] WANG, W., AND VERGYRI, D. The use of word n-grams and parts of speech for hierarchical cluster language modeling. In *Proceedings of the IEEE International Conference on Acoustic, Speech and Signal Processing* (May 2006).

[17] ZAWAYDEH, B., STALLARD, D., AND MAKHOUL, J. Guidelines for transcribing Arabic dialects. Tech. rep., Linguistic Data Consortium, 2002