Applying Finite State Morphology to Conversion Between Roman and Perso-Arabic Writing Systems

Jalal Maleki, Maziar Yaesoubi, Lars Ahrenberg

Department of Computer and Information Science
Linköping University
SE-581 83 Linköping, Sweden
jma@ida.liu.se, maziar.yaesoubi@gmail.com, lah@ida.liu.se

Abstract

This paper presents a method for converting back and forth between the Perso-Arabic and a Romanized writing systems for Persian. Given a word in one writing system, we use finite state transducers to generate morphological analysis for the word that is subsequently used to regenerate the orthography of the word in the other writing system. The system has been implemented in XFST and LEXC.

Introduction

We present a method for converting between two scripts for Persian: the traditional Perso-Arabic writing system [1] [2] and a Romanized script called Dabire [3]. The conversion system, which is being developed using Xerox LEXC and XFST tools [4], uses finite state transducers for modeling analysis and production of word forms, phonological alternations and orthographical conventions. Although our implementation is specific to Persian spoken in Iran, the orthographical conversion model is general and can be applied to any language with multiple scripts.

The essence of our approach is as follows. Let M_1 and M_2 denote morphological analysis transducers for two possible scripts of a language and let L denote a transducer that implements a stem lexicon mapping stems from one script to the other. Ideally, we can construct a script conversion transducer by composing these transducers: $M_1^i \otimes L \otimes M_2$. Here M_1^i denotes the inverse of M_1 and $\otimes$ is the operation for transducer composition.

Persian (an Indo-European language) is mainly written in variations of the Perso-Arabic script (PA-Script) [2] [5]. The Latin script was officially and berifly used in Tajikistan in the early days of the Soviet republic but was quickly abandoned in favor of the Cyrillic script [6]. Nowadays, however, the Latin script is used extensively in text-based mobile and electronic communication.

The extensive amount of information published on the Internet in PA-Script and varieties of Latin-based script motivates our work in bridging the gap by trying to understand the relationship between these scripts and also automatically converting between them. Our final goal is to create a platform for applications such as, multi-script chat, search, data-mining, and indexing for libraries.

Script Conversion Problems

In this section we will list a number of problems that complicate the script conversion task. After presenting a brief description of the PA-Script, we will list the challenging issue and indicate which problems we are concerned with here.

We are interested in converting between two different writing systems for Persian. First, the traditional PA-Script used in Iran, which is an extension of the Arabic script and includes some Persian-specific graphemes and some minor revisions to the orthographic rules of the Arabic script. The second writing system we use in our implementation is a Latin-based phonemic transcription called Dabire that is described in [3]. Since the correspondence between Persian phonemes and graphemes of Dabire is straightforward and the conventions of the script are similar to other Latin-based scripts, we will not discuss it in any detail. We will, however, give a short description of the traditional PA-Script below.

PA-Script is a semi-cursive writing system in which words are written from right to left by joining the appropriate graphemes. The typed variations of the writing system simulate the hand-written semi-cursive style and inherit its properties. The correspondence between consonants and graphemes representing them is relatively straightforward, whereas vowel representation is more complicated. Table-1 shows how the short vowel /e/, for example, may be represented in various contexts. The diacritics ـِ, ـَ, ـُ can be used to indicate the presence of a short vowel (*e*, *a*, *o* respectively), ـْ is used to indicate absence of a vowel and ـّ is used to indicate gemination. However, these diacritics are usually not used unless there is a pedagogical reason for including them. Here is an expression with three words: یک سیب قرمز (a red apple) which in the fully-vocalized version would be written as یِکْ سِیْبِ قِرْمِزْ (*yek sib e qermez*).

/e/	Word Initial	Segment Initial	Segment Medial	Segment Final	Intra-Word Solo
V, VC, VCC	ا ابراهیم	ئـ سوئز	ـئـ مطمئن	ـئه اوپئه	ئه نیکاراگوئه
CVC, CVCC		ـِـ آبِستن	ـِـ بکِش	ـِـ کدِر	
CV		ـِـ خواهِش	ـِـ بکِشید	ـه پروانه	ه آتشکده

Table 1. *Mapping /e/ to PA-Script Graphemes.* اوپئه is the transliteration of the Hawaiian word *'Opae* (shrimp).

An alphabetic word is written as a sequence of one or more segments written from right to left. Segments are separated by a zero-width space. Here we use the word *segment* to refer to a sequence of conjoined graphemes. A segments is only a graphical notion and does not necessarily represent a phonological or morphological unit.

The cursive nature of the writing system necessitates multiple allographs for a grapheme. An allograph is essentially the adaptation of a grapheme so that it can properly join its neighboring allographs. There are four different positions in which a cursive allograph can appear: *Segment-Initial*, *Segment-Medial*, *Segment-Final* and *Isolated*. Some graphemes have four allographs one for each position. Others do not join their successors and this essentially means that the grapheme either appears on its own or it ends a segment. The graphemes ا, د, ذ, ر, ز, ژ and و are semi-cursive and never join the following grapheme and therefore terminate the segment in which they appear.

The rest of this section discusses some of the problematic issues related to conversion of scripts.

Analysis Problems: PA-Script

One major problem in the analysis process of real world texts is related to tokenization. Megerdoomian [7] gives a fair account of tokenization problems in processing Persian text and suggests some remedies. One major problem is that word bounderies are not always marked correctly. Words ending in semi-cursive graphemes are not delimited properly, for example, the sentence "کار را کرد و رفت" (Did the work and left) may be written as "کارراکردورفت" without any spaces between the five words constituting it. The reason for the latter being readable at all is that all words end with semi-cursive graphemes that do not join to their successors. This is usually not a problem for the human eye familiar with the script, but to an automatic tokenizer the latter form would appear as a single token. Another example is when constituents of a complex token are separated with a normal space rather than with the zero-width non-joining space (ZWNJ) [8] which is the correct delimiter for separating orthographical segments. For example, پروانه (butterfly) and وار (like) can be joined to form the compound word پروانه‌وار (like a butterfly) where the two words are correctly separated using a ZWNJ character. However, a less carefully typed version (پروانه وار) may use space rather than a ZWNJ as separator giving the impression that we have two separate tokens.

Another issue in analysis is that Persian verbs have two stems: *present stem* and *past stem*. The present stem can in principle be derived from the past stem, see [9] for an implementation of this derivation process. However, since the number of verbs is limited, one can represent the present and the past stems separately as in [7]. Another complication in the analysis process is the existence of so the called long-distance dependencies in verbs [7].

Generation Problems: PA-Script

Given the morphological information about a word (a stem and a set of feature tags), some of the main problems in generating a PA-Script word involve vowel representation, representation of phonological alternations and also generation of ZWNJ space in compound words.

PA-Script has a relatively ad hoc set of conventions for writing compound words which allows a large number of exceptions. These are presented in a recent publication by the Persian Academy [1]. Some authors dispute the adequacy and the accuracy of

these conventions [10]. However, the main principle is to write compounds in a semi-open format[1] to make sure that the graphic identities of the sub-words of a compound are preserved as much as possible in order to minimize ambiguities. In computer-based texts, a ZWNJ-space is used to separate the constituents of a compound in order to override the cursive nature of the orthography. In contrast to PA-Script, Dabire has a simple set of conventions [3] for writing compound that clearly indicate when words should be written in open or closed format. In short, just like some European languages such as Swedish, the default format for writing compound words in Dabire is the closed format, whereas, the preferred format in PA-Script is the semi-open format.

In PA-Script, some graphemes have multiple roles, for example, ه (He) is used for denoting /h/ as well as word final /a/ and /e/. Here are some examples:

[*kuh*, کوه, **kwh**, *kuh*, mountain]
[*kuce*, کوچه, **kwch**, *kutʃɛ*, alley]
[*na*, نه, **nh**, *næ*, no]

When such a word forms the non-final sub-word of a compound token, the ه being fully-cursive can join the initial grapheme of the following word and its shape will change from the segment-final form to the segment-medial. Since ه represents a vowel only if it occurs in the segment-final or isolated form, changes in its shape may create ambiguities. It is therefore, fine to write the plural of کوه as کوهها but it is not good practice to write the plural of کوچه as کوچهها. It should be written as کوچه‌ها. Similarly, کوچه ای (an alley) is clearly a better choice than کوچهای.

The Implementation

Our system is implemented using Xerox LEXC and XFST [4] and currently consists of: a LEXC-module for specifying morphology for Dabire, a similar LEXC-module for PA-Script,[2] a syllabification method implemented in XFST (see [12]), a simple transducer that implements a lexicon for stems in Dabire and PA-Scriptand finally the main XFST-module that integrates the whole system and contains miscellaneous transducers such as alternation rules, ZWNJ-space insertion rules.

Although finite state transducers are bidirectional, the design of our morphology transducers is based on the direction of word generation, which defines how words can be constructed by systematically attaching morphological features to word stems. The complete finite state transducer for converting from Dabire to PA-Script is defined as a multi-level composition of transducers as shown in Figure-1. The left part of the figure implements the PA-Script-morphology transducer (M_1), and the right part the Dabire-morphology transducer (M_2). The box at the bottom of the diagram is a simple transducer (L) that maps between stem transcriptions.

[1] We call this format *semi*-open to distinguish it from the open format in English that uses space to separate parts of the compound [11].

[2] The LEXC-modules for the two scripts are very similar and it is possible to generate one from the other using the lexicon transducer, but we have not exploited this possibility yet.

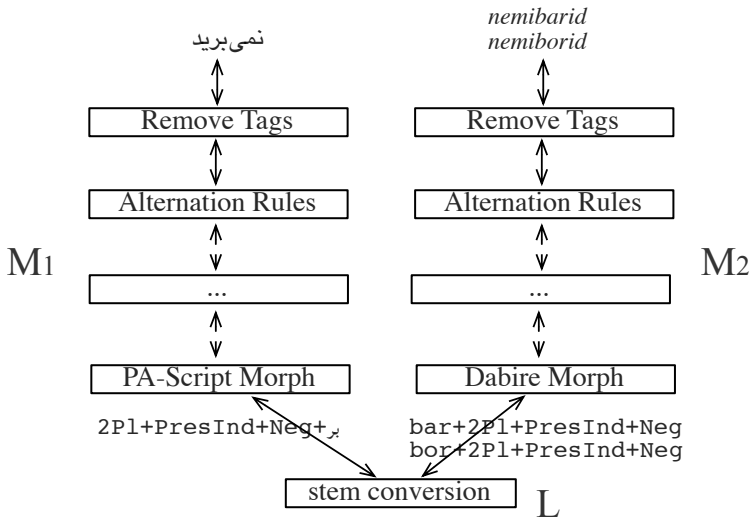

Fig. 1. The composition of transducers that use finite state morphological analysis and a stem transcription lexicon (simple transducer) to transcribe words of Dabire to PA-Scriptvice versa.

A we mentioned in the introduction, an orthographical conversion system can be subsequently defined as the composition of these transducers as follows: $M_2^i \otimes L \otimes M_1$. Here M_2^i denotes the inverse of M_2 and $\otimes$ is the operation for transducer composition [4].

The following example trace shows how the verb نمی‌برید (take+2Pl+PresInd+Negative) is analyzed to the present stem بر of the verb بردن (to take). In the trace, نمی‌برید and بر are shown as "nmybryd" and "br" respectively in order to improve readability. The steps in the trace are numbered as `M1*` and `M2*` illustrate various stages in the analysis process in M_1 and M_2 transducers.

```
M11. br+2Pl+PresInd+Neg
M12. br+2Pl+PresIndˆDur+Neg
M13. [br]+2Pl+PresIndˆDur+Neg
M14. nmy[br]+2Pl+PresIndˆDur+Neg
M15. nmy[br]yd+PresIndˆDur+Neg
M16. nmybryd
```

Dabire-morpology produces similar trace,

```
M21. bar+2Pl+PresInd+Neg
M22. bar+2Pl+PresIndˆDur+Neg
M23. [bar]+2Pl+PresIndˆDur+Neg
M24. nemi[bar]+2Pl+PresIndˆDur+Neg
M25. nemi[bar]id+PresIndˆDur+Neg
M26. nemibarid
```

It is clear from these examples that inverting either M_1 or M_2 together with a transducer for stem conversion (stem dictionary) enables us to construct a FST for converting from one writing system to another.

Finally, the implementation constitutes a relatively large number of transducers that implement the rules and conventions of the writing system. For example, the rule `i -> [a y] || .#. _` would replace word-initial occurrences of *i* with `a y` which at a later stage is transliterated to ای (see Table-1). This particular rule covers one instance of the orthography of *i* where occurs in syllables of the form V, VC, VCC and would be written independent of other segments (for example, in کارخانه‌ای (a factory)).

Finally, as an example illustrating pecularities of Persian orthography in our XFST implementation we include part of the rules for inserting zero-width spaces in the context of compound words in PA-Script. In the following rules, ZWNJ is shown as `+Z`:

```
define paZWNJ [
   [..] -> %+Z ||
        %+Pre [[? - CmpndTag ]* - [ b h | b y | h m ]]
        _ [CmpndTag - %+Pre]
    .o.
   [..] -> %+Z || %+Num  [? - CmpndTag]* _ [CmpndTag]
    .o.
   [..] -> %+Z || b _  CmpndTag b
    ...]
```

The first rule states the convention that PA-Script-prefixes other than به, بی and هم (shown as `bh`, `by` and `hm` in the rule) should not join the rest of the word [1]. The second part implements another orthographic convention of PA-Script that suggests that numbers initiating a compound word, should be separated from the rest of the word using a ZWNJ-space, for example,

[*panjzel'i*, پنج ضلعی, **pnj-ʒlʿy**, *pændʒzɛlʔɪ*, pentagon]

is a compound built using [*panj*, پنج, **pnj**, *pændʒ*, five] and [*zel'i*, ضلعی, **ʒlʿy**, *zɛlʔɪ*, sided].

Finally, the third rule which is the first instance of a series of replace rules (one for each consonant) indicates that if one constinuent of a compound ends with the same grapheme that initiates the following sub-word, then the graphemes should be separated by a zero-width space. For example, نیک (good) which ends with a "k" and کردار (deed) which starts with a "k" can join to form a compound that can either be written as نیککردار or نیک کردار. However, the latter is preferred since it discourages the reader from inserting a vowel after the first word. Essentially, this sort of complications is the price the PA-Script has to pay for continuing to avoid short vowel representation.

Evaluation

Our system has not been evaluated in a real setting mainly because our stem lexicon is very small and a lot of lexicographic work still remains. Furthermore, the system does

not cover all paradigms completely. In particular, we have not implemented the code for handling adverbs and complex verbs.

In a limited evaluation experiment, we randomly selected 448 words from Tehran University Bijankhan Corpus [13] (which uses PA-Script), included the necessary stems in the stem lexicon and tested the system in the conversion direction from PA-Script to Dabire. The system failed to analyze 13 words and successfully converted 351 words (78 percent) to Dabire without over-generation. For the remaining 104 words, the coversion was successful but there were some meaningless words among the over-generated answers.

Currently, a number of factors limit the accuracy of our system. We list them below:

Incomplete Lexicon: The lexicon we are currently using is very small, however, improvements in this respect are only a matter of time. Our ongoing work involves lexicographic extensions to the system to cover more root words.

Incomplete Orthographic Rules: Our orthographic rule-base is not complete. Although a lot of writing conventions and alternations are covered, additions and fine-tuning of the rule base is still an ongoing effort. For example, when converting from Dabire to PA-Script, rules for generating a zero width space in certain compounds is still incomplete.

Over-Generation: One of the problems we are currently experiencing is over-generation. For example, the Persian word نبرد has a number of alternative analyses and can, therefore, be Romanized as *nabard* (fight+Noun+Sg), *nabord* (take+Past+3Sg), *nabarad* (take+Pres+3Sg) or *naborad* (cut+Pres+3Sg). However, in some cases over-generation involves production of non-words. Currently we do not deal with these issues.

Limitations of XFST: Some of the inaccuracies in our system are due to the limitations of the particular version of XFST we have used. Since there is no pattern matching capabilities or ways of remembering things in the pure finite state methodology, we have not been able to implement a number of orthographic conventions that involve counting. For example, one way of building compounds in Persian is to repeat a word. The adjective *kam* (less, little) can be repeated to build the word *kamkam* (gradually, little by little). In PA-Script *kam* is written as کم and once it is repeated it can be either written as کمکم or کم کم. The latter is preferred, since the former suggests another pronunciation *komakam* (my help). However, this sort of problems can be easily managed in a pre-morphological stage which we have not addressed yet.

Conclusion

In this paper, we have briefly described a general approach to the problem of automatic conversion between two alternative scripts of a language. The main idea presented here is to generate a morphological analysis for a word written in one writing system and then use the analysis to produce the orthography for the word in the other writing system. The case of Romanized and Perso-Arabic writing systems for Persian is specially interesting since the writing systems are very different and enjoy different writing conventions.

The core of our implementation consists of finite state transducers for representing morphological analysis and production, phonological alternations and orthographical

conventions of the scripts. The system is implemented using Xerox LEXC and XFST tools [4] [14].

Although FSM-technology has been extensively used in many applications, our use of the technology for automatic transcription between multiple scripts for a single language (Persian for example) is rather unique. Related work includes [15] [16] where XFST-technology is applied to Arabic transcription and transliteration, [7] has applied XFST to morphological analysis of Persian and extended it to analysis of blogs [17]. Unfortunately, we have not been able to build upon these earlier systems since their work is proprietary.

Our future work involves extending the system to cover all morphological paradigms and a very large lexicon. Furthermore, we intend to handle words that are not represented in the lexicon. We have used a syllabification-based approach for converting correct Dabire-words that lack lexical representation [12]. We are also working on a system that uses HMM-techniques similar to [18] for converting PA-Script-words which lack lexical representation.

References

1. Farhangestan: Dastur e Khatt e Farsi (Persian Orthography). Volume Supplement No. 7 , February 2000. Persian Academy, Tehran (February 2003)
2. Neysari, S.: A Study on Persian Orthography - (in Persian). Sâzmân e Câp o Entešârât (1996)
3. Maleki, J.: A Romanized Transcription for Persian. In: Proceedings of Natural Language Processing Track (INFOS2008), Cairo. (2008)
4. Beesley, K.R., Karttunen, L.: Finite State Morphology. CSLI Publications (2003)
5. Adib-Soltâni, M.S.: An Introduction to Persian Orthography - (in Persian). Amir Kabir Publishing House, Tehrân (2000)
6. Perry, J.P.: A Tajik Persian Reference Grammar. Brill (2005)
7. Megerdoomian, K.: Finite-state morphological analysis of persian. In Farghaly, A., Megerdoomian, K., eds.: Proceedings of the Workshop on Computational Approaches to Arabic Script-based Languages. (2004) 35–41
8. Esfahbod, B.: Persian computing with unicode. In: 25th Internationalization and Unicode Conference, Washington, DC. (2004)
9. Ziai, R.: Finite state methods applied to verbal inflection in persian. Master's thesis, Eberhard-Karls Universitet, Tübingen (2006)
10. Malek, R.R.Z.: Qavâed e Emlâ ye Fârsi. Golâb (2001)
11. Ritter, R.M.: The Oxford Guide to Style. Oxford University Press (2002)
12. Maleki, J., Ahrenberg, L.: Converting Romanized Persian to Arabic Writing System Using Syllabification. In: Proceedings of the LREC2008, Marakech. (2008)
13. Bijankhan, M.: Bijankhan Corpus - Tehran University. (2008)
14. Kaplan, R.M., Kay, M.: Regular models of phonological rules systems. Computational Linguistics **20(3)** (1994) 331:378
15. Beesley, K.R.: Romanization, transcription and transliteration (1996)
16. Beesley, K.R.: Arabic finite-state morphological analysis and generation. In: Proceedngs of COLING'96 Copenhagen. (1996)
17. Megerdoomian, K.: Extending a persian morphological analyzer to blogs. In: Proceedings of the Second Workshop on Persian Language and Computers. (2006)
18. Gal, Y.: An HMM approach to vowel restoration in Arabic and Hebrew. In: Proceedings of the ACL-02 workshop on computational approaches to semitic languages. (2002) 1–7