A COMPREHENSIVE ROMAN (ENGLISH)-TO-BANGLA TRANSLITERATION SCHEME

Naushad UzZaman (naushad@bracuniversity.ac.bd), Arnab Zaheen (arnab@bracuniversity.ac.bd) and Mumit Khan (mumit@bracuniversity.ac.bd)

Center for Research on Bangla Language Processing
BRAC University, Bangladesh

Abstract:
A transliteration scheme from Roman (English) to Bangla can help increase the use of Bangla in essential and diverse computing areas such as word processing, Internet and mobile communication and information query and retrieval. The Bangla script's irregular phonetic nature and its large repertoire of consonant clusters (juktakkhors) create a large gap between the pronunciation and the orthography for a given Bangla word. In this paper, we describe a comprehensive Roman (English)-to-Bangla transliteration scheme that is designed to handle the full complexity of the Bangla script. We apply a phonetic encoding scheme to produce intermediate code-strings that facilitate matching pronunciations of input strings and the desired outputs. We also provide graceful degradation to a more conventional direct phonetic mapping in special circumstances. A prototype of our scheme shows significant success in test cases.

Key Words: Transliteration, English, Bangla, Bengali, Phonetic mapping

1. INTRODUCTION

Transliteration, in a narrow sense, is the mapping of letters from one writing script into another writing script. Ideal transliteration is loss-less, i.e., the informed reader should be able to reconstruct the original spelling of unknown transliterated words [1]. In most transliteration schemes, the letters of the source script are pronounced similarly as the letters of the goal script [2]. Transcription, on the other hand, is the system of writing the *sounds* of a word in one language using the script of another language. If the relations between letters and sounds are similar in both languages, transliteration may almost be the same as transcription. In a broader sense, the word 'transliteration' can be used to denote both transliteration in the narrow sense and additionally, transcription [1].

The scarcity of software products with native "out of the box" support for Bangla creates a significant barrier to the language's use in written forms of desktop computing such as word processing and Internet and mobile communication such as electronic mail, chatting, etc. Creating a usable transliteration scheme from Roman (English) to Bangla provides a solution to this problem, but the complex orthographic rules in Bangla pose a challenge. The Bangla script's irregular phonetic nature and its large repertoire of consonant clusters (*juktakkhors*) create a large gap between the pronunciation and the orthography for a given Bangla word.

Based on this knowledge, we introduce in this paper a comprehensive Roman (English)-to-Bangla transliteration scheme that handles the full complexity of the Bangla script with the assistance of a phonetic lexicon. In our proposed transliteration scheme, there are two types of mapping: a direct phonetic mapping and a lexicon-enabled phonetic mapping. With direct mapping we implement transliteration in the narrow sense — a lossless mapping from the source to the goal language script. With phonetic lexicon-enabled mapping, on the other hand, we realize transliteration in the broader sense, i.e., we make it simultaneously work as a transcription system.

2. PREVIOUS WORK

Building transliteration schemes for English (Roman) to non-European languages is a significant research challenge, as demonstrated by the plethora of activities involving English to Japanese [3], English to Arabic [4-7] and English to Chinese [8]. These transliteration schemes are also used in various applications such as cross language information retrieval using statistical analysis. Bangla, despite being the fourth most widely spoken language [9], does not yet have a comprehensively defined transliteration scheme. The most noteworthy work on transliteration from Roman (English) to Bangla has been implemented in ITRANS [10, 11] in early 1991. Some other established transliteration schemes include ISO 15919 [12] and Harvard-Kyoto [13]. Currently there are also a few word processors that support transliteration of Bangla using the Roman script [15-19]. All these schemes only support direct phonetic mapping and are consequently loss-less. However, these schemes are not able to handle the complex cases in applications like cross language information retrieval.

3. DIRECT PHONETIC MAPPING

We consider direct mapping a trivial phonetic mapping scheme that maps letters from the source script to the goal script without the help of a phonetic lexicon. Existing transliterations from Roman (English) to Bangla use this method. The most popular such mapping is provided in ITRANS [11], which is used by a number of applications [5, 6], while others have defined their own. In this paper, we provide another direct mapping method to be used in our own transliteration scheme.

There is a key difference between existing English-to-Bangla direct phonetic mapping methods and ours. In traditional direct mapping schemes, the user is provided with only one letter or letter-group in the source script to represent one letter in the goal script, i.e., these are one-to-one mapping schemes. In our direct phonetic mapping scheme, for a number of suitable cases, we provide the user with multiple options of such letter or letter-groups in the source script (which, in our case, is Roman) to represent one letter in the goal script (Bangla), i.e., ours is, partially, a many-to-one mapping scheme. We realize this flexibility by introducing an intermediate step where the input-string in the source script is converted to an intermediate code-string before its final conversion into the goal script. For example, the Roman-letter source input strings *phul, phool, fool* and *ful* should all correspond to the word ফুল /p^{h}ul/ in Bangla. In our method, each of these four input-strings corresponds to just one intermediate code-string <phul>, which is finally converted to the corresponding Bangla word ফুল.

Table 1 completely describes our proposed rules for direct phonetic mapping from Roman (English) to Bangla. It includes the valid input character/character-groups in the source script and their corresponding intermediate and final output forms. The Unicode number of each Bangla letter is also given.

Table 1. Table for direct phonetic mapping

Roman letter or letter-group	**Intermediate encoding**	**Name**	**Bangla letter**	**Unicode**
a	a	AA	আ	\u0986
	a	SIGN AA	া	\u09BE
b	b	BA	ব	\u09AC
bh	bh	BHA	ভ	\u09AD
c/ch	c	CA	চ	\u099A
Ch/chh	ch	CHA	ছ	\u099B
d	d	DA	দ	\u09A6
dh	dh	DHA	ধ	\u09A7
D	D	DDA	ড	\u09A1
Dh	Dh	DDHA	ঢ	\u09A2
e	e	E	এ	\u098F
	e	SIGN E	ে	\u09C7
f	ph	PHA	ফ	\u09AB
g	g	GA	গ	\u0997
gh	gh	GHA	ঘ	\u0998

h	h	HA	হ	\u09B9
H	H	VISARGA	ঃ	\u0983
i	i	I	ই	\u0987
	i	SIGN I	ি	\u09BF
I	I	II	ঈ	\u0988
	I	SIGN II	ী	\u09C0
j	j	YA	য	\u09AF
J	J	JA	জ	\u099C
jh	jh	JHA	ঝ	\u099D
k	k	KA	ক	\u0995
kh	kh	KHA	খ	\u0996
l	l	LA	ল	\u09B2
m	m	MA	ম	\u09AE
M	M	CANDRABINDU	ঁ	\u0981
n	n	NA	ন	\u09A8
N	N	NNA	ণ	\u09A3
Nh	Nh	NYA	ঞ	\u099E
ng	ng	ANUSVARA	ং	\u0982
Ng	Ng	NGA	ঙ	\u0999
o	o	A	অ	\u0985
O @ BEGIN	O	O	ও	\u0993
O @ MIDDLE/END	O	SIGN O	ো	\u09CB
oi	oi	AI	ঐ	\u0990
	oi	SIGN AI	ৈ	\u09C8
ou	ou	AU	ঔ	\u0994
	ou	SIGN AU	ৌ	\u09CC
oo	u	SIGN U	উ	\u09C1
p	p	PA	প	\u09AA
ph	ph	PHA	ফ	\u09AB
q	k	KA	ক	\u0995
r	r	RA	র	\u09B0
R	R	RRA	ড়	\u09DC
Rh	Rh	DDHA	ঢ়	\u09A2
s	s	SA	স	\u09B8
sh	sh	SHA	শ	\u09B6
S	S	SSA	ষ	\u09B7
t	t	TA	ত	\u09A4
th	th	THA	থ	\u09A5
T	T	TTA	ট	\u099F
Th	Th	TTHA	ঠ	\u09A0
u	u	U	উ	\u0989
	u	SIGN U	ু	\u09C1
U	U	UU	ঊ	\u098A
	U	SIGN UU	ূ	\u09C2

v	bh	BHA	ভ	\u09AD
w	u	UU	ঊ	\u098A
x @ BEGIN	j	YA	য	\u09AF
x @ MIDDLE/END	ks	KA SA	কস	\u0995 \u09B8
y	y	YYA	য়	\u09DF
z	j	YA	য	\u09AF
\	\	HASANT	◌্	\u09CD

4. LEXICON-ENABLED PHONETIC MAPPING

A good transliteration scheme between two languages enables the application of cross language information retrieval and has been successfully implemented in many languages. Most of these implementations, however, require extensive statistical analyses of source and goal languages. There is a dearth of such analyses in Bangla at the moment. An alternate solution to this problem is to use a lexicon with phonetic code generation capability. Such a lexicon can be easily implemented for Bangla, allowing cross-lingual information query and retrieval until sufficient statistical data is available.

The first step in building a usable phonetic lexicon is to generate a phonetic code-string for each Bangla word from a supplied lexicon. However, extracting the code-strings for Bangla words of similar pronunciation (homonyms) is not easy in Bangla, mostly due to the language's complex orthographic rules. One solution described in the literature uses a Double Metaphone encoding scheme for Bangla in various applications such as a spelling checker or a word similarity checker [20, 21]. We use a slightly modified and improved version of the Double Matephone encoding scheme for our purposes. The following section shows the algorithm.

4.1. Algorithm for phonetic lexicon-enabled mapping

```
1. Generate the phonetic code-strings for all Bangla words in the supplied lexicon using the
modified and improved Double Metaphone encoding scheme and store these as items in a phonetic
lexicon.
2. Input Bangla word using Roman (English) characters.
3. Generate the phonetic code-string of the input.
4. IF the input's phonetic code-string matches the phonetic code-string corresponding to only
one word in the Bangla lexicon THEN

   convert the input to that Bangla word.

ELSE IF the input's phonetic code-string matches a phonetic code-string corresponding to
multiple words in the Bangla lexicon THEN
   produce suggestions of all relevant Bangla word outputs and let the user select the correct
   output.

ELSE IF the input's phonetic code-string does not match with any available phonetic code-
string in the phonetic lexicon THEN
   convert the input to Bangla using direct mapping.
```

The challenge here is to generate the phonetic code-strings of the Bangla words in the lexicon as well as the phonetic code-strings for the Roman input strings. Additionally, any major inconsistencies we encounter while matching these two types of strings need to be removed.

4.2. Generating phonetic code-strings of Bangla words

As mentioned earlier, we use the Double Metaphone phonetic encoding scheme for Bangla proposed in [20, 21] to encode the words in our Bangla lexicon into corresponding phonetic code-strings. We convert the Roman character input-string in a similar manner. For example, we describe in this section how the Bangla word কলম is encoded into <klm> using the phonetic encoding rule of [20, 21]. Our other challenge is to describe how to encode the Roman (English) input in a way so that when someone writes *kolom*, it is also converted to <klm> (this will be discussed in section 4.3).

4.2.1. Modifications done to Double Metaphone encoding for the purpose of our scheme

The phonetic encoding scheme we used to convert Bangla words is a slightly modified version of the one proposed in [20, 21]. The modifications were necessary to make the mapping consistent from both directions

(Roman input and Bangla lexicon entry).

4.2.1.1. The case of aspirated consonants

We made some modifications to the encoding proposed in [20, 21] to distinguish aspirated consonants from their unaspirated counterparts.

Table 2. Modification to encoding in [20, 21]

Bangla letter	Name	Unicode	Encoding in [20]	Modified encoding
ভ	BHA	\u09AD	"b"	"bh"
ছ	CHA	\u099B	"c"	"ch"
ধ	DHA	\u09A7	"d"	"dh"
ঢ	DDHA	\u09A2	"d"	"dh"
ঘ	GHA	\u0998	"g"	"gh"
ঝ	JHA	\u099D	"j"	"jh"
খ	KHA	\u0996	"k"	"kh"
ফ	PHA	\u09AB	"p"	"ph"
থ	THA	\u09A5	"t"	"th"
ঠ	TTHA	\u09A0	"T"	"Th"

4.2.1.2. The case of Ya-phalaa following aspirated consonants

While the general accuracy of the Double Metaphone phonetic encoding proposed in [20, 21] is supported by statistics, we had to make some minor improvements to get the best out of the method. In [20, 21], when the Bangla letter য /ya/ (YA) appears as the latter constituent in a consonant cluster in the middle or end of a word in a post-base form (as in পদ্য/podă̤o/ or বাধ্যতা /badd^{h}ŏ̤a/), the encoding doubles the pronunciation of the preceding constituent of the cluster. For example, for the words কাব্য /kabbo/ and পদ্য/podă̤o/, য /ya/ (Unicode YA) is converted to the same phonetic code as the preceding constituent of the cluster. In কাব্য /kabbo/, য /ya/ (Unicode YA) will be converted to <b>, which is the code of ব /b/, and the resulting phonetic code-string will be <kabb>. Similarly for পদ্য /podă̤o/, য /ya/ (YA) will be converted to <d>, the code for দ /d̤/, and the ensuing phonetic code-string will be <pdd>. Although this method (of replacing য /ya/ with the previous constituent consonant of the cluster) works very well in these examples, it fails when the consonant preceding য /ya/ is aspirated. For example, তথ্য /tŏ̤t^{h}o/, বাধ্য /bad̤̆^{h}o/, সখ্য /ʃokk^{h}o/, and লভ্য /lobb^{h}o/ will have phonetic codes <tthth>, <badhdh>, <skhkh>, and <lbhbh>, respectively. These code-strings clearly do not express the correct pronunciations of the corresponding Bangla words. Instead of a plain doubling of the aspirated consonant, the consonant cluster is correctly pronounced as a combination of un-aspirated and aspirated consonants, e.g, ধ্য is pronounced not as ধ+ধ (/d̤^{h}/+/d̤^{h}/) but as দ+ধ (/d̤/+/d̤^{h}/). We therefore handle this problem by modifying our algorithm so that it revises such code-strings to reflect this pronunciation rule, producing <ttth>, <baddh>, <skkh>, and <lbbh> respectively for the four examples we considered.

4.2.1.3. The case of ambiguity concerning Reph+YA and RA+YA-phalaa

The phonetic encoding in [20, 21] is based on Unicode and there was an ambiguity concerning the use of Bangla Reph and Ya-phalaa until Unicode 4.0 [23]. According to [23], the Unicode format defines that Reph is formed when a RA (র), which has the inherent vowel killed by the virama/halant, begins a syllable. This is shown in the following example:

র + ◌্ + ম ⟶ র্ম as in কর্ম (karma)

The YA-phalaa (্য) is a post-base form of YA (য) and is formed when the Ya is the final consonant of a syllable cluster. In this case, the previous consonant retains its base shape and the virama/halant is combined with the following Ya. This is shown in the following example.

ক + ◌্ + য ⟶ ক্য as in বাক্য

An ambiguous situation is encountered when the combination of Ra + virama/halant + Ya is encountered.

র + ্ + য ⟶ র্য or র‍্য

To resolve the ambiguity with this combination and to have consistent behavior, the latest Unicode standard takes into account the processing order of the Bengali script. When parsing the text, the ability to form the Reph is identified first and therefore the Reph form should have priority in processing. Thus, it is necessary to insert a U+200C ZERO WIDTH NON-JOINER character (ZWNJ in short) into the stream between the Ra and virama/halant to allow the virama/halant and Ya to be grouped together during processing:

র + ্ + য ⟶ র্য
(U+09B0 + U+09CD + U+09AF)

র + ZWNJ + ্ + য ⟶ র‍্য
(U+09B0 + U+200C + U+09CD + U+09AF)

In the example above, the ZWNJ is used because two characters that would join by default are intended to remain as separate entities. In cases other than where the RA is the first character in the cluster, the ZWNJ is not required for the formation of the Ya-phalaa. However, for ease of placing the Ya-phalaa input as a single key input, it should be permissible for the Ya-phalaa to be consistently formed by "ZWNJ + VIRAMA + YA" (U+200C + U+09CD + U+09AF).

It is clear that there is an ambiguity in writing the character sequence র+্ +য in Bangla and that its solution has been given in Unicode 4.0.1 [23]. However, the software we use to implement transliteration uses previous versions of Unicode and is unable to handle this case at present. In the future, we will be able to resolve this ambiguity by employing a ZWNJ and produce different codes for different cases. In the meantime, we propose a temporary solution as described below. In our encoding scheme, we had problems with this ambiguity specifically when the sequence র+্ +য occurred in the middle or the end of a word (e.g., সূর্য /ʃurjo/, আশ্চর্য্যান্বিত /aʃchorjanniṭo/). In Bangla language, the sequence র+্ +য always appears as র্য in the middle or the end of a word; there is no case of র‍্য in those circumstances[24]. We have, therefore, modified our algorithm to only consider র্য when encountered with র+্ +য in the middle or the end of a word.

4.3. Generating phonetic code-strings of Roman (English) word inputs

In Table 3, we propose the phonetic encoding scheme for Roman character input-strings. We use almost the same direct mapping scheme from Table 1, with different intermediate codes for a few cases. Table 3 contains only those exceptions. We also termed “Intermediate encoding” of Table 1 “Encoding like Bangla” in Table 3.

Table 3. Proposed encoding for phonetic mapping

English letter or bigram	**Encoding like Bangla**	***Name***	**Bangla letter**	**Unicode**
H	h	VISARGA	ঃ	\u0983
J	j	JA	জ	\u099C
M	Not Coded	CANDRABINDU	ঁ	\u0981
N	n	NNA	ণ	\u09A3
Ng	ng	NGA	ঙ	\u0999
O	Not Coded	A	অ	\u0985
O @ BEGIN	o	O	ও	\u0993
O @ MIDDLE/END	Not Coded	SIGN O	ো	\u09CB
R	r	RRA	ড়	\u09DC
Rh	r	DDHA	ঢ়	\u09A2
Sh	s	SHA	শ	\u09B6
S	s	SSA	ষ	\u09B7
\	Not Coded	HASANT	্	\u09CD

5. EXAMPLE OF TRANSLITERATION

We now show the outputs of both direct and phonetic mapping schemes for a given Roman (English) text input.

Following is a Bangla text input using English alphabet:

```
ami bhalo achi. tomar khobor ki. ajke shondha bela tumi ki korcho. obak bepar holo, ami ekhon bangla likhte pari inglish diye. aro mojar bepar holo ami dui bhabe likhte pari. ekTa DairekT arekTa phoneTik. tomar desh e koto Taka te ek Dolar. ami ei bhabe abar juk\to bor\no likhte pari.
```

5.1. Output in direct mapping

Our output in direct mapping for the above input will be the following:

আমি ভালো আছি. তোমার খোবোর কি. আযকে শোনধা বেলা তুমি কি কোরছো. অবাক বেপার হোলো, আমি এখোন বাংলা লিখতে পারি ইনশলিশ দিয়ে. আরো মোযার বেপার হোলো আমি দুই ভাবে লিখতে পারি. একটা ডাইরেকট আরেকটা ফোনেটিক. তোমার দেশ এ কোতো টাকা তে এক ডোলার. আমি এই ভাবে আবার যুক্তো বোর্নো লিখতে পারি.

5.2. Output in phonetic mapping

And the output using phonetic mapping will be the following:

আমি বহাল/ভাল/ভালো আছি. তোমার খবর কই/কি/কী. আজকে সন্ধ্যা বেলা তুমি কই/কি/কী করছ. অবাক বেপার/ব্যাপার হল, আমি এখন/এখনো বাংলা/বাঙলা লিখতে পারি/পাড়ি ইংলিশ দিয়ে. আর/আরো/আড় মজার বেপার/ব্যাপার হল আমি দুই ভাবে লিখতে পারি/পাড়ি. একটা ডাইরেকট আরেকটা ফোনেটিক . তোমার দেশ/দ্বেষ এ কত/কোঁত টাঁকা/টাকা তে এঁকো/এক ডলার . আমি এই ভাবে আবার যুক্ত বরণ/র্বণ/ব্রণ লিখতে পারি/পাড়ি

Table 4. Few examples from above paragraph to make the process clear

English word	Output in direct mapping	Output in phonetic mapping	Selected word
shondha	শোনধা	সন্ধ্যা	সন্ধ্যা
bela	বেলা	বেলা	বেলা
bepar	বেপার	বেপার/ব্যাপার	ব্যাপার
mojar	মোযার	মজার	মজার
DairekT[1]	ডাইরেকট	ডাইরেকট	ডাইরেকট
ami	আমি	আমি	আমি
ek	এক	এঁকো/এক	এক
juk\to	যুক্তো	যুক্ত	যুক্ত
bor\no	বোর্নো	বরণ/র্বণ/ব্রণ	র্বণ

Table 4 shows how we can handle the similar sounding multiple words in a suggestion. We can select our expected word among the suggestions either manually or generate the correct word automatically, given available contextual statistical data and appropriate methods.

6. PERFORMANCE

A transliteration software-prototype [25] was implemented based on the methods discussed above and tested with users. The users were given an introduction on how to use the software. Even though the main idea behind this phonetic mapping is that the user can write in the Roman character(s) based on his (ideally competent) knowledge of Bangla pronunciation, there are letters in Bangla that can cause ambiguity as they are usually written using the same Roman character. For example, ত /t̪/, থ /t̪^{h}/, ট /ʈ/, ঠ /ʈ^h/ — all of these letters are written using the Roman character "t" or "th". We asked the user to use specific Roman characters for a few specific Bangla characters, which means there remains an irreducible element of direct phonetic mapping for a few Bangla letters even when we are using phonetic mapping with a phonetic lexicon. For the example cases

[1] Not found in the lexicon, so direct mapping used for the suggestion

mentioned above, we instruct the users to use the following codes: "t" for ত /t̪/, "th" for থ /t̪ʰ/, "T" for ট /t/, and "Th" for ঠ /tʰ/.

We encountered a couple of problems during this try-out: i) for a given input-string the corresponding word is entirely absent from the lexicon, ii) the inflected form of a head-word is sometimes missing from the lexicon, e.g., we may have a সরকার /ʃɔrkar/ in our lexicon, but we may not have other inflected forms of সরকার /ʃɔrkar/ such as সরকারের /ʃɔrkarer/, etc.

We handle these cases by providing a graceful degradation from lexicon-enabled phonetic mapping into direct phonetic mapping, as described in the last ELSEIF clause in the 4th step of our algorithm presented in section 4.1. The phonetic lexicon for our prototype. In our performance checking survey, more than 10 users took approximately 2500 words from Bangla newspaper articles and inputted them in Roman (English) format. We found that 32% of the input strings did not have corresponding Bangla words in our lexicon. Graceful degradation of these cases to direct phonetic mapping provided the correct output string in 23% of the cases. We could not handle the remaining words (9% of total) mainly because of the limitations of our current lexicon, which already contains more than 100,000 entries of Bangla words. This limitation can be overcome in two ways: i) by increasing the number of words in our lexicon and ii) by using morphological synthesis to efficiently generate all possible inflected words for the lexicon. Currently, our scheme provides the user with the correct output in 100% of the cases where the relevant Bangla word is present from the lexicon.

Performance Statistics at a glance:

Words found in the lexicon: 68%

Given the word is in the lexicon, the instances it was handled properly by phonetic mapping with phonetic lexicon: 100%

Words not found in the lexicon: 32%

Words not found in the lexicon but handled properly by direct mapping: 23%

Words not found in the lexicon because of the absence of inflected words: 7%

Words not found in the lexicon and not handled properly by direct mapping: 2%

7. USING TRANSLITERATION WITH PHONETIC MAPPING IN CROSS LANGUAGE INFORMATION RETRIEVAL APPLICATION

In cross language information retrieval, a user issues a query in one language to search a collection in a different language. If the two languages use the same alphabet then similar sounding words can be written in the same way in two languages and can easily be found as well. However, if two languages use two different alphabets then it is not an easy task to issue a query in one language to search a collection in a different language.

A cross language information retrieval application can be developed using our proposed transliteration scheme with phonetic lexicon-enabled phonetic mapping. In this application, the input will be a Roman character input-string and it will retrieve similarly pronounced Bangla words from Bangla documents. Details of this application can be found in [20].

8. CONCLUSION

We have designed a phonetic lexicon-based English-to-Bangla transliteration (and, simultaneously, transcription) scheme that is more comprehensive than the schemes realized so far by others. Our proposed transliteration scheme is meant to act as a bridge until a more thorough computational linguistic appraisal of the Bangla language is realized. Our scheme can also relieve desktop and mobile-device users of the burden to learn multiple Bangla-based input methods for different systems and devices. In addition, the scheme can be used in powerful applications such as cross language information query and retrieval.

9. ACKNOWLEDGEMENT

This work has been supported in part by the PAN Localization Project (www.panl10n.net), grant from the International Development Research Center, Ottawa, Canada, administrated through Center for Research in Urdu Language Processing, National University of Computer and Emerging Sciences, Pakistan. We would also like to thank other members of our research group and BRAC University students who helped by participating in our performance survey.

10. REFERENCES

[1] Wikipedia entry of Transliteration, available online at http://en.wikipedia.org/wiki/Transliteration

[2] Wikipedia entry of Transcription, available online at http://en.wikipedia.org/wiki/Transcription_%28linguistics%29

[3] Kevin Knight and Jonathan Graehl, "Machine Transliteration", Computational Linguistics 24(4): 599-612 (1998).

[4] Yaser Al-Onaizan and Kevin Knight, "Machine Transliteration of Names in Arabic Text", Proc. of ACL Workshop on Computational Approaches to Semitic Languages, 2002.

[5] Nasreen Abdul Jaleel and Leah S. Larkey, "English to Arabic Transliteration for Information Retrieval: A Statistical Approach", CIIR Technical Report IR-261, Dept. of Computer Science, University of Massachusetts.

[6] Leah S. Larkey, Nasreen Abdul Jaleel, Margaret Connell, "What's in a Name?: Proper Names in Arabic Cross Language Information Retrieval", CIIR Technical Report, IR-278.

[7] Nasreen Abdul Jaleel and Leah S. Larkey, "Statistical Transliteration for English-Arabic Cross Language Information Retrieval", CIKM 2003: Proceedings of the twelfth international conference on information and knowledge management , New Orleans, LA, 139-146.

[8] GAO Wei, "Phoneme based Statistical Transliteration of Foreign Names for OOV problem", MSc Thesis, Chinese University of Hong Kong, 2004.

[9] The Summer Institute of Linguistics (SIL) Ethnologue Survey 1999, available online at http://www2.ignatius.edu/faculty/turner/languages.htm.

[10] ITRANS, available online at http://www.aczoom.com/itrans/

[11] ITRANS table, available online at http://sanskrit.gde.to/web-interface/bengali.html

[12] ISO 15919 Transliteration of Devanagari and related Indic scripts into Latin characters, available online at http://homepage.ntlworld.com/stone-catend/trind.htm.

[13] Harvard-Kyoto Convention for transliterating the Sanskrit language in ASCII, available online at http://en.wikipedia.org/wiki/Harvard-Kyoto.

[14] Aksharmala mapping, available online at http://aksharamala.com/help/chm/Input%20Schemes/ITRANS/Bengali/quick.html

[15] Iwrite32, available online at http://members.tripod.com/~sbiswas/IWrite32/IWrite32.html

[16] Bornosoft, available online at http://www.bornosoft.com/

[17] Kickkeys, available online at http://www.kickkeys.com/

[18] Lekho, available online at http://lekho.sourceforge.net/

[19] Bengali Transliteration System by prabashi.org, available online at http://www.prabasi.org/Literary/ComposeArticle.html

[20] Naushad UzZaman, "Phonetic Encoding for Bangla and its Application to Spelling checker, Transliteration, Cross language information retrieval and Name searching", Undergraduate thesis (Computer Science), BRAC University, May 2005.

[21] Naushad UzZaman and Mumit Khan, "A Double Metaphone Encoding for Bangla and its Application in Spelling Checker", Proc. 2005 IEEE Natural Language Processing and Knowledge Engineering, Wuhan, China, October, 2005.

[22] Definition of phonetic encoding available online at http://www.nist.gov/dads/HTML/phoneticEncoding.html.

[23] Clarification of Bengali Reph and Ya-phalaa in Unicode 4.0.1, available online at http://www.unicode.org/versions/Unicode4.0.1/.

[24] Dr. Suniti Kumar Chatterji, "Bhasha-Prakash Bangala Vyakaran", D. Mehra Publishers, Kolkata, May 1989, Page 59

[25] Prototype of Transliteration available online at http://student.bu.ac.bd/~naushad/software/pata/