Implementing Urdu Grammar as Open Source Software

اردو

Muhammad Humayoun

Department of Mathematics
University of Savoie
France
mhuma@univ-savoie.fr

Harald Hammarström
Aarne Ranta

Department of Computer Science
Chalmers University of Technology
Sweden
{harald2, aarne}@cs.chalmers.se

CLT 07, University of Peshawar
Pakistan

UNIVERSITE CHAMBERY ANNECY DE SAVOIE

Introduction

- ***Indo-European → Indo-Iranian → Indo-Aryan***
- Written from right to left using Perso-Arabic Script.
- Grammar and Vocabulary influenced by Arabic, Persian and the native languages of South Asia
- Widely Spoken in Pakistan, Jammu & Kashmir and India
- Also spoken all over the world due to big South Asian Diaspora
- Urdu-Hindi: share grammar, almost all phonology and lot of vocabulary
- Urdu-Hindi together is the second most widely spoken language (Native + second language)

Contribution

- **Orthography component**: A Unicode Infrastructure to accommodate Perso-Arabic script of Urdu
- **Morphology component** :
  - A type system that covers the language abstraction completely
  - An inflection engine that covers word-and-paradigm morphological rules for all word classes
- **Lexicon**: Automatically extracted, 4,816 words generating 137,182 word forms.
- **Grammar component** : A small fragment of syntax

Plan

- Urdu Orthography
- Urdu Morphology
- The Lexicon
- Urdu Syntax
- A complete example
- Conclusion & Future Work

Plan

- Urdu Orthography
- Urdu Morphology
- The Lexicon
- Urdu Syntax
- A complete example
- Conclusion & Future Work

Urdu Orthography

- An alphabet of 57 letters and 15 diacritic marks
- The use of diacritic marks: optional
- Morphology and the lexicon saved in ASCII characters
  - Reusability for Hindi in future, by adding a lexicon and the transliteration scheme
  - Easy manipulation on different platforms

Urdu Orthography

- Transliteration scheme (Transliterator)
  - Unicode value of each letter is mapped with a unique ASCII string
  - Clear, strict and reversible
- Useful tools
  - Urdu Extractor
  - Keyboard input method
- A GUI application
- Implemented in Java by using ICU4J and Swing packages

Urdu Orthography - Transliteration

```
public class UrduUnicode
{
   public static final char alif = '\u0627';
   public static final char bay = '\u0628';
   public static final char pay = '\u067e';
  …..
 }
```

```
public class UrduRoman
{
   public static final String alif = "a";
   public static final String bay = "b";
   public static final String pay = "p";
   ….
}
```

```
private static final String unicode_to_Roman_rules =
            UrduUnicode.alif     +  ">" +  UrduRoman.alif       +  ";" +
            UrduUnicode.bay      +  ">" +  UrduRoman.bay        +  ";" +
            ……
```

```
public static final Transliterator unicode_to_roman =
Transliterator.createFromRules("RomanUrdu-Unicode", unicode_to_Roman_rules, 0);
```

```
String romanText =
Transliterator_ur.unicode_to_roman.transliterate("Unicode Text");
```

Urdu Orthography - Transliteration

Examples:

کِتاب	Kiṯab	k(i)tab	book
کَوشِش	koshish	k(a)wX(i)X	struggle
بُلاؤ	bulaʔʊ	b(o)law^	to call

ک	→	k	ـَ	→	(a)
ـِ	→	(i)	ش	→	X
ت	→	t	ـُ	→	(o)
ا	→	a	ؤ	→	w^
ب	→	b			

Plan

- Urdu Orthography
- Urdu Morphology
- The Lexicon
- Urdu Syntax
- A complete example
- Conclusion & Future Work

Morphology Development
Current state of art

- Special-purpose languages based on finite-state technology
- The most well-known among others: **XFST**, **LexC** **(Xerox)**
  - Implementation: regular expressions, string types
  - Complexity: linear runtime, advance compilation techniques
  - Efficiency: fast, but large networks
- Non-regular formalisms might be difficult to implement efficiently enough

Some open questions

- Does the direct coding allow to implement one's linguistic abstraction adequately?
  - Expressiveness
- Is such model extensible and reusable?
  - How much cost to add a lexical item?
  - Will refinement of information require global re-design
  - Will it cause inconsistencies
- How can it be integrated into applications
  - API and user Interfaces, Software Localization, modularity?

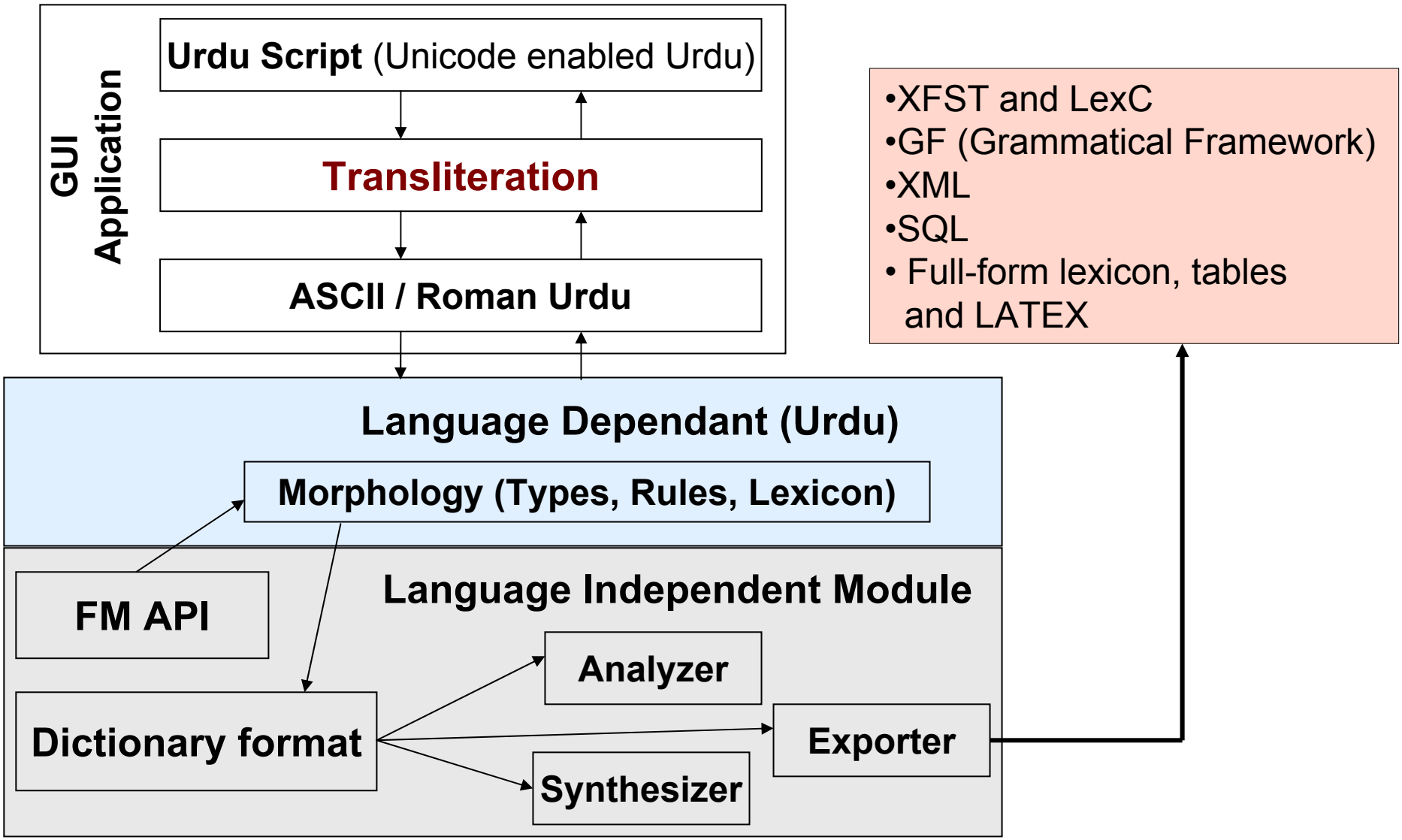
Functional Morphology

- Morphology is implemented in Functional Morphology (FM)
- An open source toolkit or domain embedded language for morphology development in Haskell
- Designed by Markus Forsberg & Aarne Ranta
  - Chalmers University of Technology, Sweden
  - Sept 2004, International Conference on Functional Programming
- Inspired by Gèrard Huet's Zen toolkit
  - Computational processing of Sanskrit, 2002
- Idea: Dealing with grammars as reusable software libraries

Functional Morphology (cont)

- Haskell: a functional programming language
  - High level of abstraction
  - Higher-order functions
  - Type classes
  - Polymorphism
  - These features: good for capturing linguistic generalizations
- Functional Morphology treats
  - The part of speech (word classes) as **data types**
  - Their Inflection as **finite functions**
- Tools (API functions, Analyzer, Synthesizer, Exporter)

Morphology + Orthography



Functional Morphology Toolkit

Nouns in Urdu: type system

- Urdu Noun Inflects in **Number** *(Singular, Plural)* and **Case** *(Nominative, Oblique, Vocative)*

```
data Number      =  Singular | Plural
                 deriving (Show, Eq, Enum, Ord, Bounded)

data Case        =  Nominative | Oblique |  Vocative
                 ....

data NounForm    =  NF Number Case
                 ....

type Noun        =  NounForm → Str
                 ....
```

- Inherent parameter: **Gender** *(Masculine, Feminine)*

```
data Gender      = Masculine | Feminine
                 deriving (Show,Eq,Enum,Ord,Bounded)
```

Nouns in Urdu

- Nouns are divided into 15 groups based on their inflection
- Masculine & Feminine proper names: 1 group for each
- A group as running example:
  - Singular masculine nouns ending with (ا, ɑ) , (ہ, h) and (ع, e / ʔ / ə)
- Making:
  - If a word ends with letter (ا, ɑ) or (ہ, h) then:
    - **Plural nominative, singular oblique**: last letter is replaced by (ے, e)
    - **Plural oblique**: the last letter is replaced by (وں, oɳ)
    - **Plural vocative**: last letter is replaced by (و, o)
  - If a word ends with (ع, e / ʔ / ə): above mentioned letters will be added at the end without replacing any existing letter

Nouns in Urdu: inflection engine

Example Noun: (laṛka, لَڑکا ,boy)

	Nominative	*Oblique*	*Vocative*
Singular	laṛka لَڑکا	laṛke لَڑکے	laṛke لَڑکے
Plural	laṛke لَڑکے	laṛkoṇ لَڑکوں	laṛko لَڑکو

```
noun_lRka :: DictForm → Noun
noun_lRka lRka nf = mkNoun  sg  pl  sg_obl  pl_obl  sg_voc  pl_voc  nf
                    where
                        sg       = lRka
                        pl       = lRk ++ "E"
                        sg_obl   = pl
                        pl_obl   = lRk ++ "wN"
                        sg_voc   = pl
                        pl_voc   = lRk ++ "w"
                        lRk      = if (end =="e") then lRka else (tk 1 lRka)
                        end      = dp 1 lRka
```

Nouns in Urdu: inflection engine

- An general function for the Inflection table of nouns

mkNoun::String→String→String→String→String→String→Number→Case → String

mkNoun sg pl sg_Obl pl_Obl sg_Voc pl_Voc n c =

```
case n of
  Singular → case c of
      Nominative   → sg
      Oblique      → sg_Obl
      Vocative     → sg_Voc
  Plural → case c of
      Nominative   → pl
      Oblique      → pl_Obl
      Vocative     → pl_Voc
```

	Nom	*Obl*	*Voc*
Sg	laɽka لَڑکا	laɽke لَڑکے	laɽke لَڑکے
Pl	laɽke لَڑکے	laɽkoɳ لَڑکوں	laɽko لَڑکو

Nouns in Urdu: inflection engine

- An interface function for this group of nouns

  n1 :: DictForm → Entry

  n1 df = masculine (noun_lRka df)

DictForm: a string type

masculine: a function for masculine words

- Defined in the Lexicon:

  n1 l(a)R'ka (lə ɽkɑ, لَڑکا)

  n1 b(o)r'q(a)e (bʊrkɑ, بُرقَع)

  n1 p(a)r'd(a)h (pərɖɑ, پردہ)

  …

Nouns in Urdu: inflection engine

- An interface function for this group of nouns

  n1 :: DictForm → Entry

  n1 df = masculine (noun_lRka df)

DictForm: a string type

masculine: a function for masculine words

- Defined in the Lexicon:

  n1 l(a)R'ka (ləɽkɑ, لَڑکا)

  n1 b(o)r'q(a)e (bʊrkɑ, بُرقَع)

  n1 p(a)r'd(a)h (pərd̪ɑ, پردہ)

  …

laɽka

N

Masc

NF Sg Nom: laɽka

NF Sg Obl: laɽke

NF Sg Voc: laɽke

NF Pl Nom: laɽke

NF Pl Obl: laɽkoɳ

NF Pl Voc: laɽko

Nouns in Urdu: inflection engine

Some other noun-groups in the Lexicon

- Singular masculine nouns ending with (ا, ɑɳ)

  n2 k(o)n'waN (Well, kʊɳwaɳ, کُنواں)

  n2 d(o)|hwaN (smoke, ḍʊɳwaɳ, دُھواں)

  ….

	Singular	Plural
Nom	kʊɳwaɳ کُنواں	kʊɳweɳ کُنویں
Obl	kʊɳweɳ کُنویں	kʊɳwoɳ کُنووں
Voc	kʊɳweɳ کُنویں	kʊɳwoɳ کُنووں

- Singular masculine nouns not ending with (ا, ɑ), (ہ, h), (ع, e / ʔ / ə) and (ا, ɑɳ)

  n3 m(a)r'd (Man, mərḍ, مَرد)

  n3 Aftab (Sun, ɑftɑb, آفتاب)

	Singular	Plural
Nom	mərḍ مَرد	mərḍ مَرد
Obl	mərḍ مَرد	mərḍoɳ مَردوں
Voc	mərḍ مَرد	mərḍo مَردو

Nouns in Urdu: inflection engine

Some other noun-groups in the Lexicon

- Singular feminine nouns ending with (ی, y)

  n4 k(o)r'sy (Chair, kʊrsi, کرسی)

- Singular feminine nouns ending with (ا, a), (اں, aɳ), (وں, oɳ)

  n5 maN (Mother, maɳ, ماں)

- Singular feminine nouns ending with (یا, yɑ)

  n6 g(o)R'ya (Doll, gʊɽiyɑ, گڑیا)

- ... n7 .... n14 ....

- Worst-case function for Noun

  n15 **singular** singular_obl **singular_voc** plural **plural_obl** plural_voc

Urdu Verbs

	Root	**Infinitive**	Oblique
Intransitive / Transitive / Ditransitive etc	bən بن	bənnɑ بننا *to build (by unknown)*	bənne بنے
Direct Causative	bənɑ بنا	bənɑnɑ بنانا *to build (by self)*	bənɑne بنانے
Indirect Causative	bənvɑ بنوا	bənvɑnɑ بنوانا *to build (by third person)*	bənvɑne بنوانے

We divide verbs in the following categories:

- Basic stem form, direct & indirect causatives exist
- Only Basic stem form exists
- Basic stem form & direct causative form exist
- Basis stem form & indirect causative form exist
- 6 groups have been implemented for verbs

Urdu Verbs

- Urdu verb inflects in:
  - Gender, Number
  - Person
    - First
    - Second {*casual, familiar, respectful*}
    - Third {*near, distant*}
  - Tense
    - Subjunctive
    - Perfective
    - Imperfective

Urdu Verbs: type system

- Category: *Basic stem form, direct & indirect causatives exist*

```
type Verb = VerbForm → Str

data VerbForm =
   VF Tense Person Number Gender                    |
   Caus1 Tense Person Number Gender                 |
   Caus2 Tense Person Number Gender                 |
   Inf          | Caus1_Inf       | Caus2_Inf       |
   Inf_Fem      | Caus1_Inf_Fem   | Caus2_Inf_Fem   |
   Inf_Obl      | Caus1_Inf_Obl   | Caus2_Inf_Obl   |
   Root         | Caus1_Root      | Caus2_Root

data Person = Pers1 |
              Pers2_Casual | Pers2_Familiar | Pers2_Respect  |
              Pers3_Near | Pers3_Distant

data Tense = Subj | Perf | Imperf
```

Urdu Verbs: inflection engine

```
mkVerbCaus12 :: String -> String -> String -> Verb
mkVerbCaus12 vInf caus1_inf caus2_inf =
    mkGenVerb root r1 r2 vInf caus1_inf caus2_inf
     where
          root      = (tk 2 vInf)
          r1        = (tk 2 caus1_inf)
          r2        = (tk 2 caus2_inf)
```

- An general function for the Inflection table

mkGenVerb::DictForm → DictForm → DictForm → DictForm → DictForm → DictForm → Verb

mkGenVerb root r1 r2 vf caus1 caus2 (Root) = root

mkGenVerb root r1 r2 vf caus1 caus2 (Inf)= vf

mkGenVerb root r1 r2 vf caus1 caus2 (Inf_Obl)= (tk 1 vf)++"E"

mkGenVerb root r1 r2 vf caus1 caus2 (Inf_Fem)= (tk 1 vf)++"y“

.....

mkGenVerb root r1 r2 vf caus1 caus2 (VF t p n g) = ***mkVAnalysis*** root t p n g

mkGenVerb root r1 r2 vf caus1 caus2 (Caus1 t p n g) = ***mkVAnalysis*** r1 t p n g

mkGenVerb root r1 r2 vf caus1 caus2 (Caus2 t p n g) = ***mkVAnalysis*** r2 t p n g

Urdu Verbs: inflection engine

```
mkVAnalysis :: String → Tense → Person → Number → Gender → String
mkVAnalysis root tense p n g =
   case tense of
      Subjunctive → case p of
            Pers1 → case n of
               Singular → case g of
                  _ → mkEnding b root "w^N" "wN“
               Plural → case g of
                  _ → mkEnding1 b root "y^yN" "yN“
            _           ->  mkSubjunctive root p n g
            where
                        t = dp 1 root
                        b = inStr t ["A","a","w"]
       Perfective → mkPerfective root p n g
       Imperfective → case p of
            Pers2_Familiar → case n of
               Singular → case g of
                  Masc → root ++ "tE"
                  Fem → root ++ "ty" ++ " " ++ root ++ "tyN"
.....
.....
```

Urdu Verbs: in lexicon

- **v4 bnna bnana bnwana**

  (bənna بننا , bənana بنانا , bənvana بنوانا)

Root: **bən** بن
Inf: **bənna** بننا
Inf_Obl: **bənne** بننے
Inf_Fem: **bənni** بنی

Caus1_Root: **bəna** بنا
Caus1_Inf: **bənana** بنانا
Caus1_Inf_Obl: **bənane** بنانے

Caus2_Root: **bənva** بنوا
Caus2_Inf: **bənvana** بنوانا
Caus2_Inf_Obl: **bənvane** بنوانے

VF Subj Pers1 Sg Masc/ Fem : **bənʊɳ** بنوں
VF Subj Pers1 Pl Masc/ Fem : **bənen** بنیں
VF Subj Pers2_Casual Sg Masc/ Fem : **bən** بن
VF Subj Pers2_Casual Pl Masc/ Fem : **bənʊ** بنو

....

Caus1 Subj Pers1 Sg Masc/Fem: **bənaʔʊɳ** بناؤں
Caus1 Subj Pers1 Pl Masc/Fem: **bənaʔen** بنائیں
Caus1 Subj Pers2_Casual Sg Masc/Fem: **bəna** بنا
Caus1 Subj Pers2_Casual Pl Masc/Fem: **bənaʔo** بناؤ

.....

Caus2 Subj Pers1 Sg Masc/Fem: **bənvaʔʊɳ** بنواؤں
Caus2 Subj Pers1 Pl Masc/Fem: **bənvaʔen** بنوائیں
Caus2 Subj Pers2_Casual Sg Masc/Fem: **bənva** بنوا
Caus2 Subj Pers2_Casual Pl Masc/Fem: **bənaʔo** بنواؤ

.....

Other word classes

- Adjectives
  - Marked (Inflects in number, case and gender)
    - Ends with (ا, ɑ): nɪlɑ نِیلا, nɪlɪ نِیلی, nɪle نِیلے : Blue
    - Ends with (اں , ɑɳ): ḍɑyɑɳ دایاں, ḍɑʔeɳ دائَیں: Right
  - Unmarked
    - No inflection : khʊsh خُوش: Happy
    - Ends with (ی,i), made from nouns or Adjectives
      - desi دیسی from noun des دیس : local
    - Inflects in degree (Persian's style)
      - bəḍ بد,bəḍṯr بدتر, bəḍṯrɪn بدترین : bad, worse, worst
- Adverbs
- The closed classes
  - Pronouns, PostPositions, Particles, Interjunctions, Conjunctions, Negations, Questions and Numerals

Plan

- Urdu Orthography
- Urdu Morphology
- The Lexicon
- Urdu Syntax
- A complete example
- Conclusion & Future Work

The Lexicon

- A wide-coverage lexicon is a key part of any morphological implementation
- Aim: to build a lexicon automatically with minimal human efforts
- A tool ***extract*** is used which is provided with the Functional Morphology
- It requires a ***paradigm file*** and a ***corpus***
- To build a corpus:
  - A reasonable amount of Urdu Unicode text was collected from the web (news and literature domain)
  - All the html tags & other non-related information were thrown away by a tool (developed with this work) and save the file as text file
  - Urdu Unicode text is then converted into ASCII Urdu by transliteration tool
- **The lexicon**: extracted by applying *paradigms* on *corpus*

The Lexicon – Example Paradigm(1)

- **Singular Feminine nouns not ending with (ا , a), (ں , N), (و , w)**

(کتاب, Kiṯab, book), (گاجر, gadʒər, carrot)

```
regexp Not_awN = char* (char- ("a" | "N" | "w"));

paradigm n9 [x:Not_awN] =
  x { (x &
        (x+"yN" | x+"wN" | x+"w")
      )
    };

      n9 k(i)tab
      n9 gajr
      n9 kRwahT
      n9 kar
      .....
```

	Nom	Oblique	Voc
Sg	کتاب Kiṯab		
Pl	کتابیں Kiṯabeɳ (yN)	کتابوں Kiṯaboɳ (wN)	کتابو Kiṯabo (w)

The Lexicon – Example Paradigm(2)

- **Verbs that has basic, direct & indirect causative forms:**

```
paradigm v4 =
  x +"na" x+"ana" x+"wana”
     {
       x+"na" & (x+"ana" | x+"wana")
     };
```

v4 dyk|hna d(i)k|hana d(i)k|hwana (ديكهنا , دِكهانا , دِكهوانا)

v4 bnna bnana bnwana (بننا ,بنانا ,بنوانا)

The Lexicon: Problems

- Urdu is commonly written without or with a variant number of diacritic marks
  - A fundamental limitation to get a fully vocalized corpus
- Problem: having more versions per word with different diacritics
  - e.g. (کِتاب, kɪ̤tɑb) and (کتاب, kt̤ɑb) for word (kɪ̤tɑb, book)
  - **Point**: We should save only one version per word with full diacritics
- Tokens with different diacritics are not always same words
  - e.g. (تَیر, t̪ær, to swim) and (تِیر, t̪ir, arrow)
  - **Point** : We should save all such words with full diacritics

The Lexicon Extraction - Results

- To assure the correctness:
  - Manually re-checking of the lexicon from word to word
  - Incorrect entries thrown away
- A fundamental limitation
  - The missing diacritics on partly vocalized words are not applied

Corpus	
Size (Words)	1,520,000 (1.5 million)
Words containing Diacritics:	23,696
Unique tokens:	63,700 (4.1%) **

Lexicon	
Extracted lexicon	9,126
Words containing Diacritics:	632
Clean lexicon	4,816 (52.8%)
Words containing Diacritics:	415

**This conforms well to our intuition that high frequent items (postpositions, auxiliaries, particles and pronouns), account for most tokens in Urdu text.

The Lexicon Extraction - Results

- Why so many incorrect entries?
- The strictness of rules in paradigm file: normal
  - Trade-off: quality vs. coverage
- Spelling mistakes:
  - Original Typos
  - Lack of spaces between words
  - Extra spaces inside words
  - Possible Reason: The use of Urdu on web is relatively new
- Foreign words:
  - Arabic – The verses of Holy Quran in religious text
  - Persian – Poetry in slightly old literary text
  - Lot of proper nouns and English words in the news domain

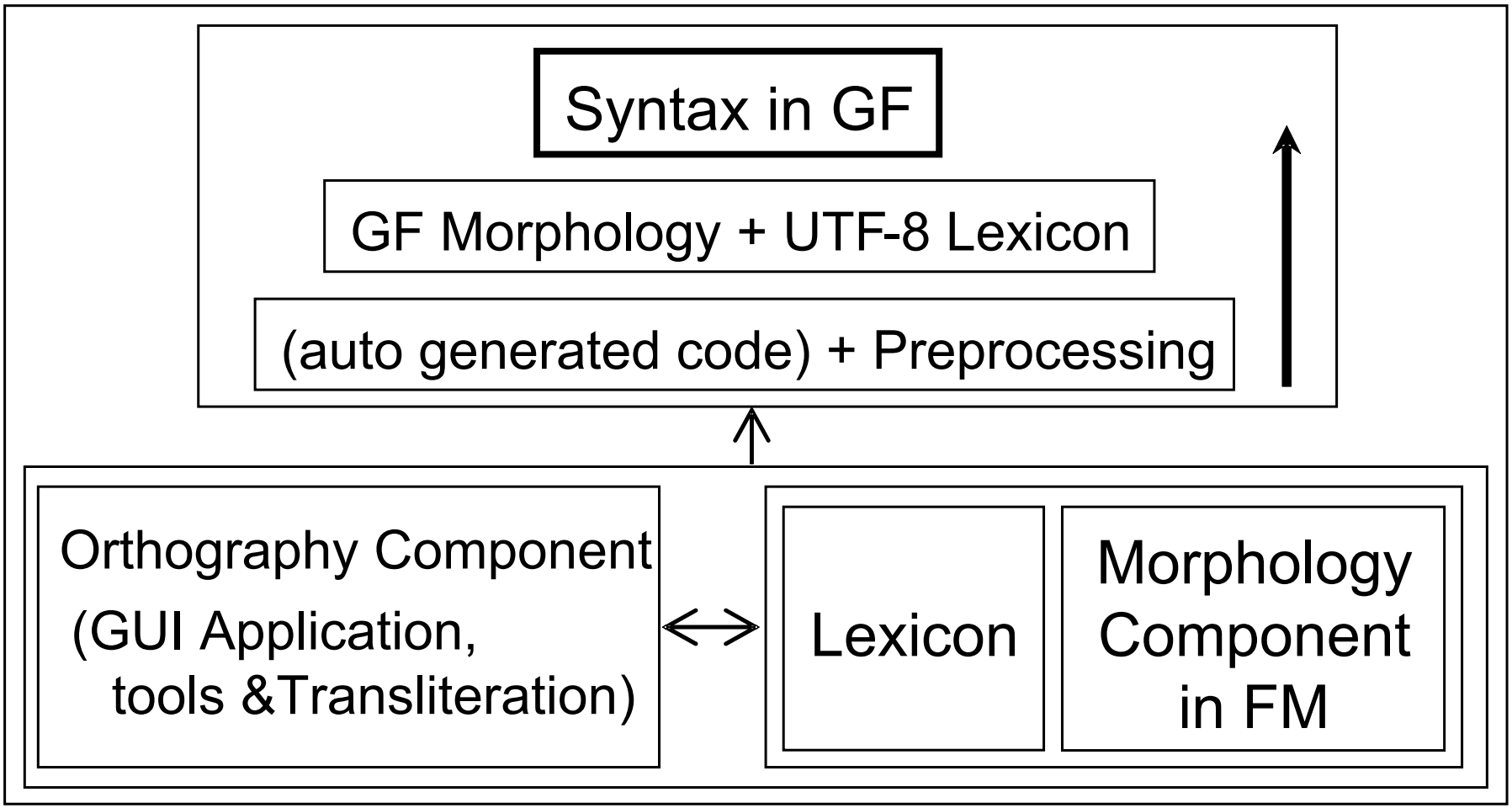
Plan

- Urdu Orthography
- Urdu Morphology
- The Lexicon
- Urdu Syntax
- A complete example
- Conclusion & Future Work

The Syntax

- Urdu an SOV (Subject Object Verb) language
- Relatively free word order
- A small fragment of syntax as a separate component on top of morphology by using Grammatical Framework
- Grammatical Framework (GF):
  - Grammar formalism based on type theory
  - Programming language for defining grammars (formal + natural)
  - Grammar = The Abstract syntax and Concrete syntax
  - Abstract syntax = semantic conditions (correct syntactic structures / trees of a language)
  - Concrete syntax = mapping abstract syntax into strings along-with the grammatical features for a language (and back, by reversibility)

The Overall Picture



The Syntax

- In our Implementation: A sentence:
  - Combination a noun phrase (NP) and a verb phrase (VP)
  - Combination of two sentences by adding a conjunction in between

The Syntax

- Abstract Syntax:

  fun UsePresS: NP → VP → S;

- Concrete Syntax:

  UsePresS np vp =

  {s = np.s ! Nom ++ vp.s ! Present ! np.p ! np.n ! np.g}

is ko kɪt̪aben leni hen, اِس کو کتابیں لینی ہیں,He/she is suppose to take the books

DemPron → Num → CN → NP	ye d̪o kɪt̪aben, یہ دو کتابیں, these two books
DemPron→ PN → NP	vo Ali, وہ علی, that Ali
NP → PostP → CN → NP	is ko kɪt̪aben, اِس کو کتابیں, to him the books

Verb_Aux → VP	hen, ہیں, are
Verb → Verb_Aux → VP	leni t̪hin, لینی تھیں, was suppose to take

More examples sentences

Plan

- Urdu Orthography
- Urdu Morphology
- The Lexicon
- Urdu Syntax
- A complete example
- Conclusion & Future Work

A Complete Example

is ko kɪt̪abeɳ leni heɳ) اِس کو کتابیں لینی ہیں

Transliteration: a(i)s kw ktabyN lyny hyN

< اِس, a(i)s >

yih_6 یِہ +DemPron - Sg Obl - Pers3_Near

mayN_8 مَیں+PersPron - Sg Pers3_Near Obl

< کو, kw >

kw_18 کو +PostP

< کتابیں, ktabyN >

ktab_824 کتاب +N - Pl Nom - Fem

< لینی, lyny >

lyna_2 لینا +Verb - Inf_Fem

< ہیں, hyN >

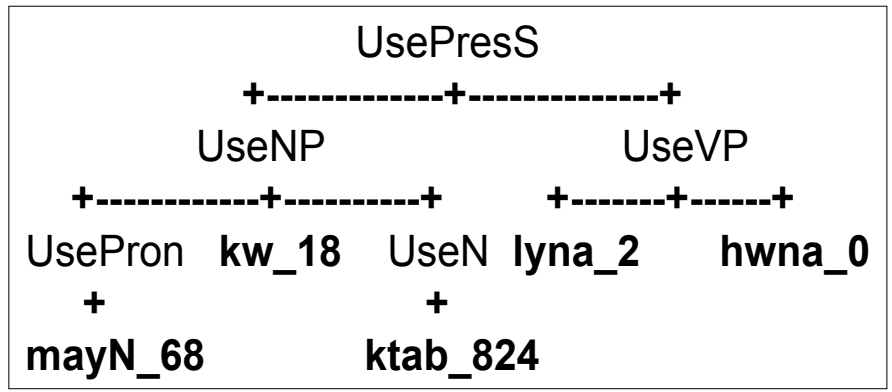
hwna_0 ہونا+Verb_Aux - Present Pers1 Pl Masc

hwna_0 ہونا +Verb_Aux - Present Pers1 Pl Fem

...

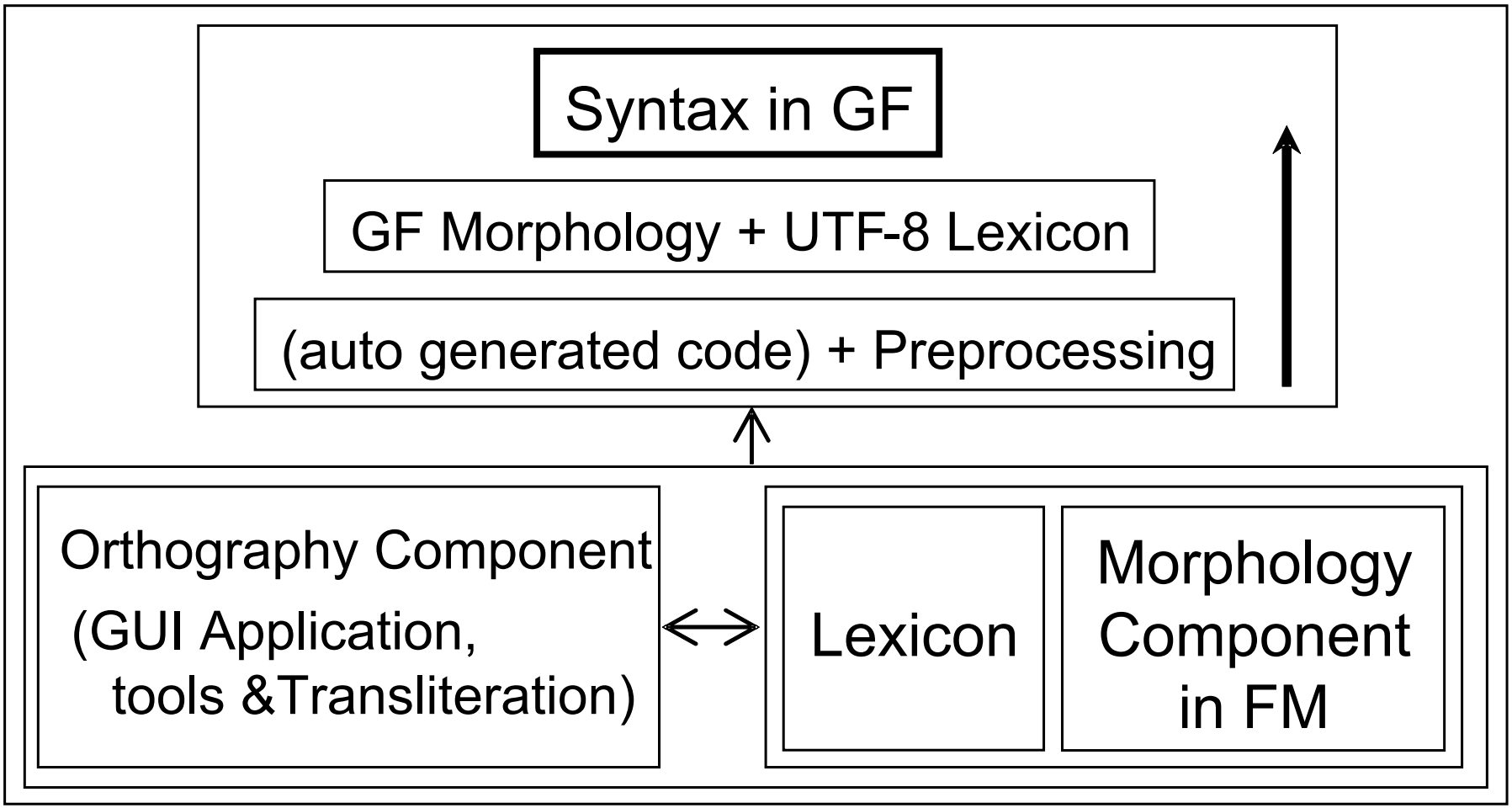
Syntactic parsing:

```
UsePresS
  (UseNP (UsePron  mayN_68)  kw_18
                             (UseN ktab_824))
  (UseVP  lyna_2  hwna_0)
```



Syntax tree

The Overall Picture



Plan

- Urdu Orthography
- Urdu Morphology
- The Lexicon
- Urdu Syntax
- A complete example
- Conclusion & Future Work

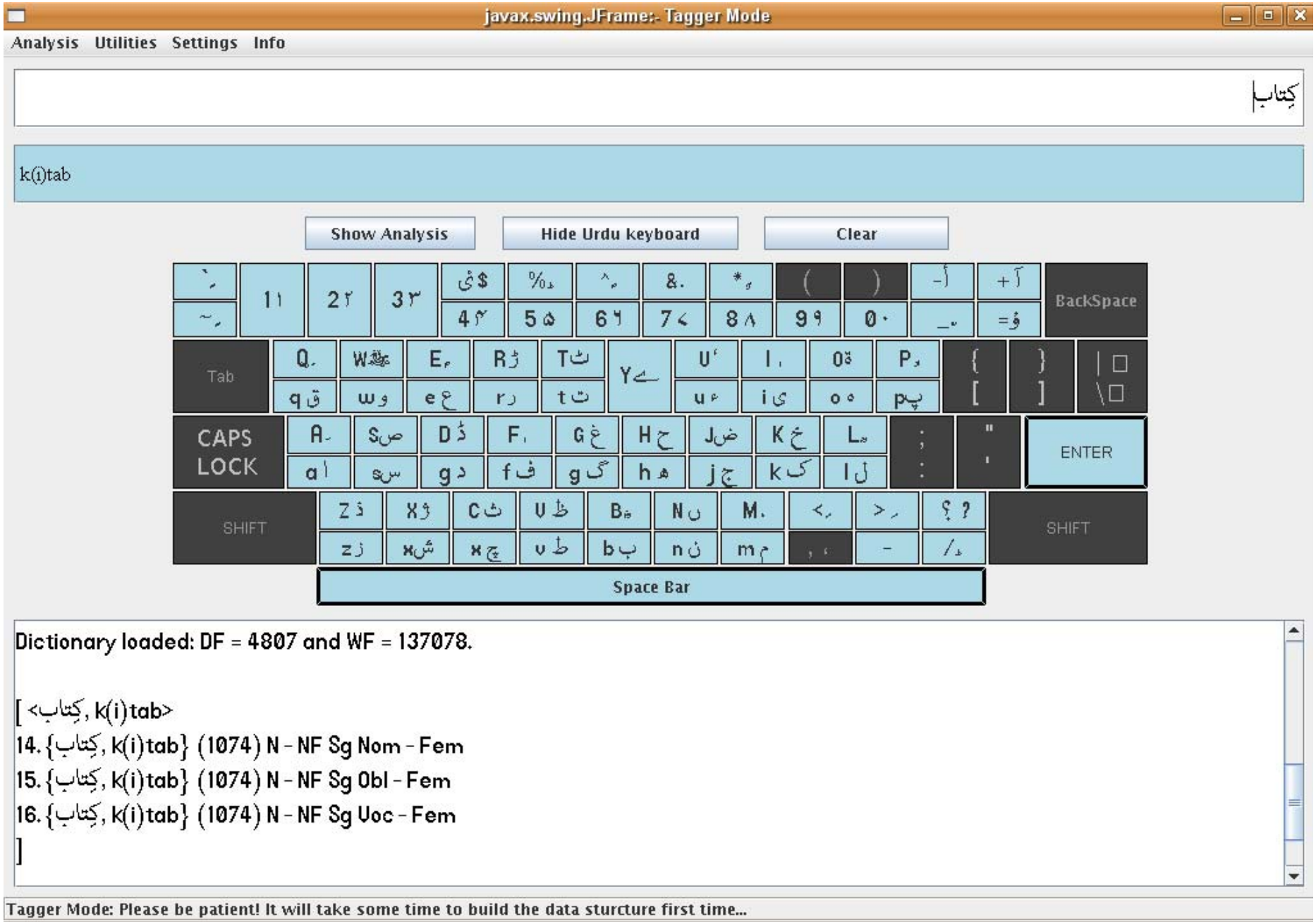
Conclusion

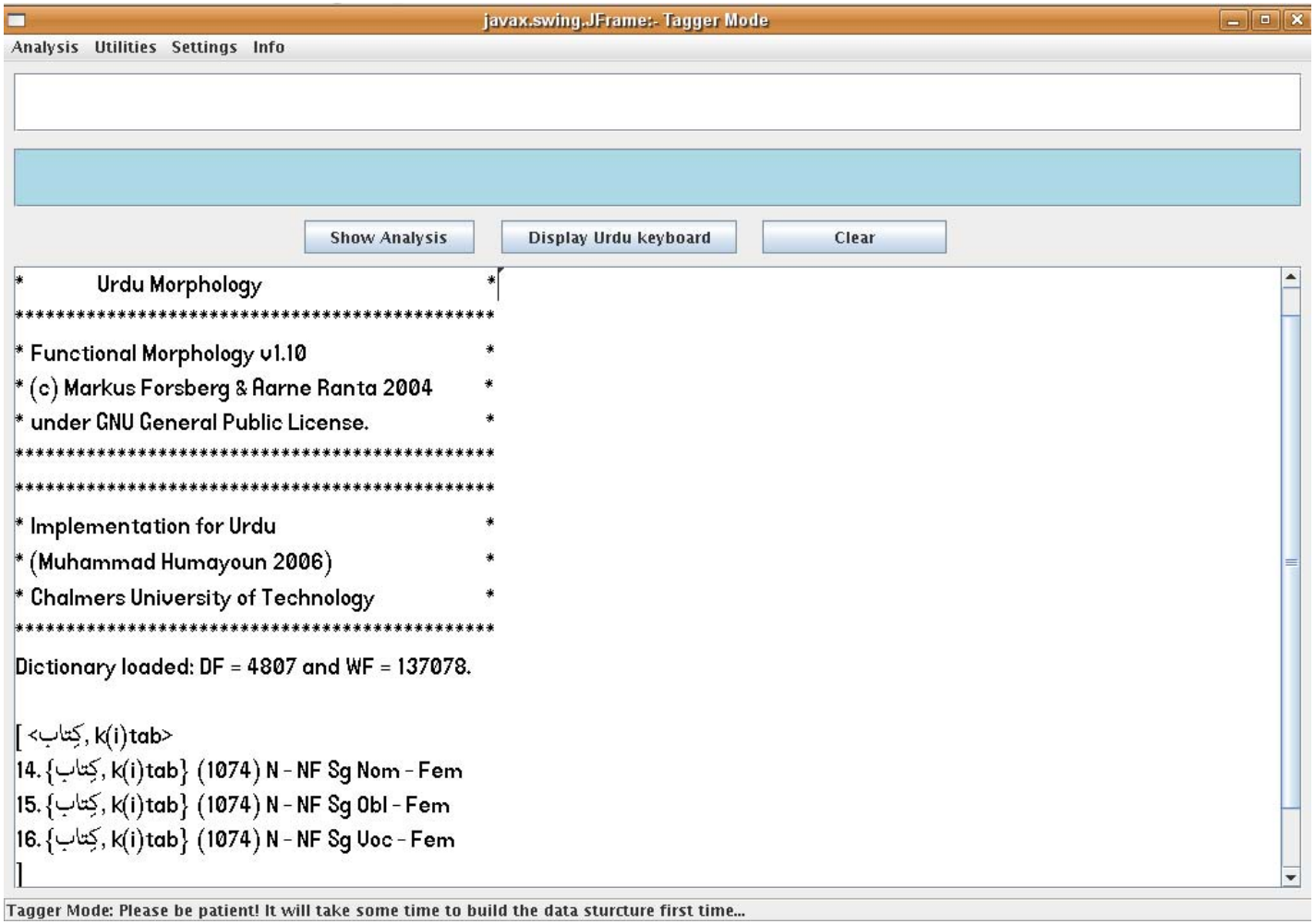
- Merits:
  - Functional Morphology proved to be a very good choice for implementing Urdu morphology
  - A *comprehensive*, *reusable* & *elegant* implementation for Urdu that covers the linguistic abstraction (morphology) adequately
- Limitations:
  - Can't deal with words that have spaces in between
  - A partly vocalized Lexicon
  - Run time system requires an exact match:
    - One cannot check if there exist orthographically different versions of a word

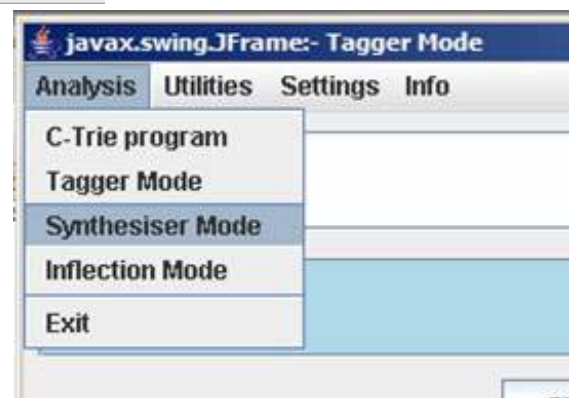
Future Work

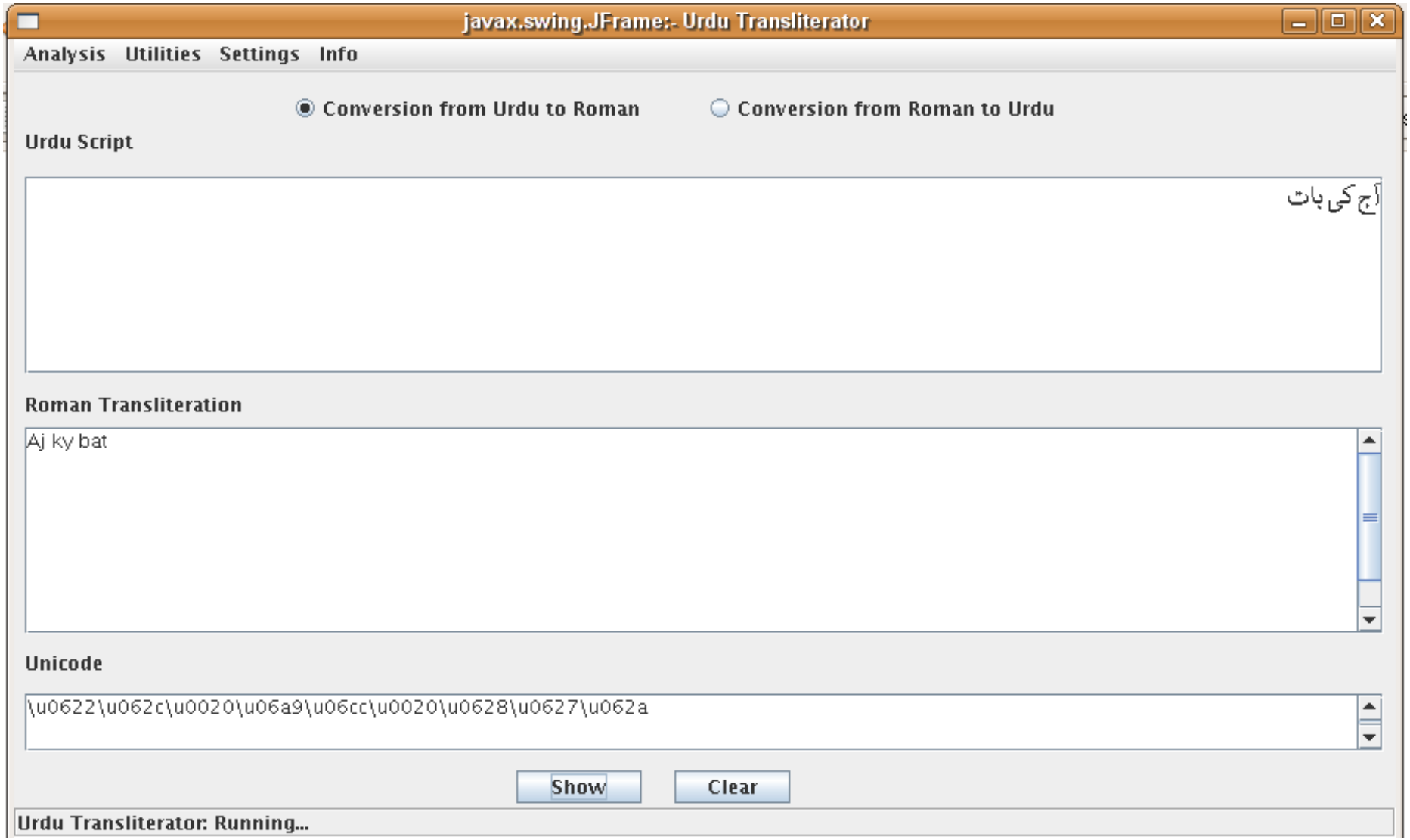
- A component that matches the partly vocalized input words with the canonical words in the lexicon
- Algorithms to add missing diacritics on partly vocalized words
- A bigger lexicon
- Analysis of words that contains spaces in between
- A comprehensive implementation for syntax
- Implementation of Hindi, by adding a lexicon and a transliteration scheme
- The Next Big Step: Urdu-English Machine Translation

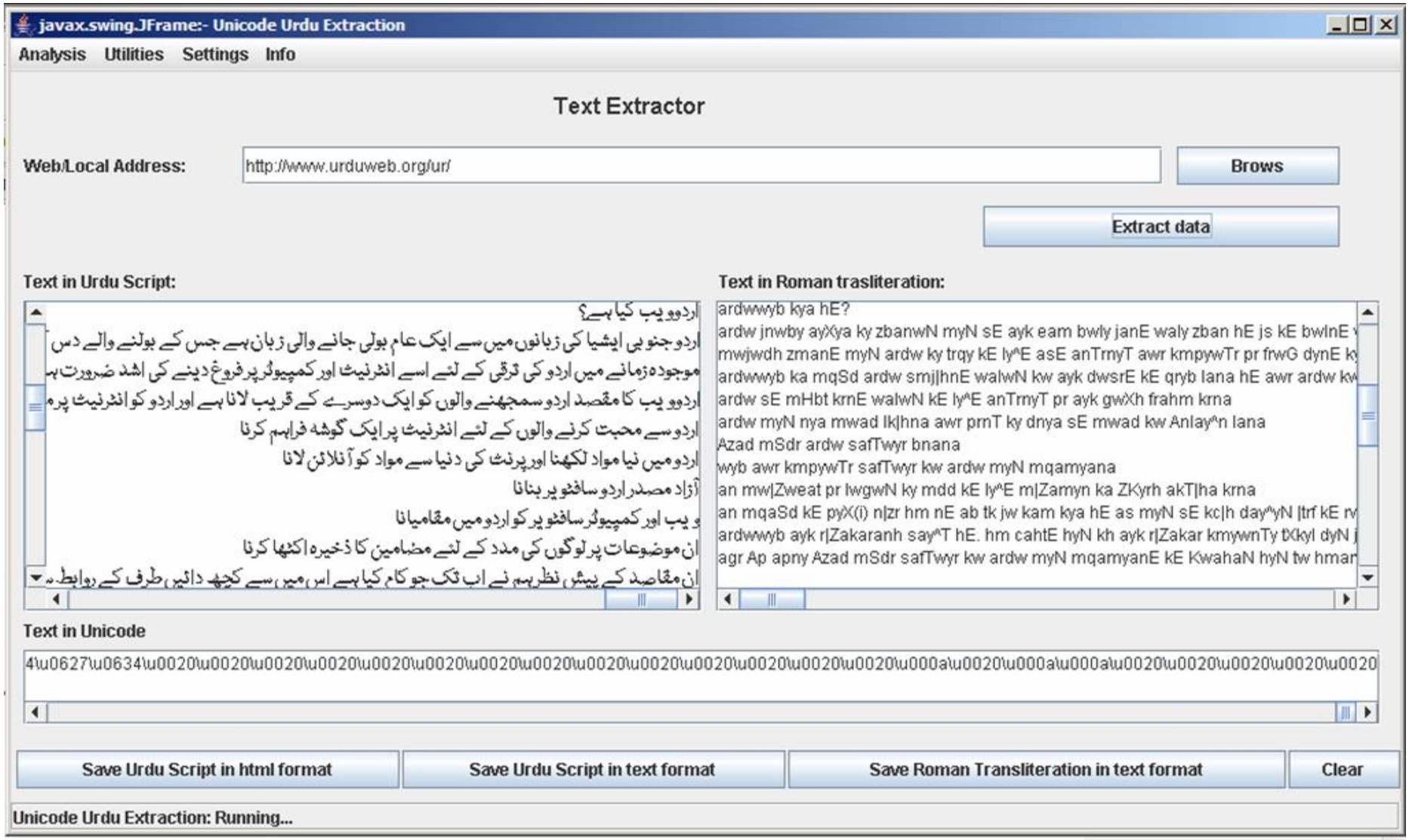
Some screen shots


javax.swing.JFrame:- Tagger Mode
Analysis Utilities Settings Info
کِتاب
k(i)tab
Show Analysis
Hide Urdu keyboard
Clear
BackSpace
Tab
CAPS LOCK
ENTER
SHIFT
SHIFT
Space Bar
Dictionary loaded: DF = 4807 and WF = 137078.
[<کِتاب, k(i)tab>
14. {کِتاب, k(i)tab} (1074) N - NF Sg Nom - Fem
15. {کِتاب, k(i)tab} (1074) N - NF Sg Obl - Fem
16. {کِتاب, k(i)tab} (1074) N - NF Sg Voc - Fem
]
Tagger Mode: Please be patient! It will take some time to build the data sturcture first time...


javax.swing.JFrame:- Tagger Mode
Analysis Utilities Settings Info
Show Analysis
Display Urdu keyboard
Clear
* Urdu Morphology *
****
* Functional Morphology v1.10 *
* (c) Markus Forsberg & Aarne Ranta 2004 *
* under GNU General Public License. *
****
****
* Implementation for Urdu *
* (Muhammad Humayoun 2006) *
* Chalmers University of Technology *
****
Dictionary loaded: DF = 4807 and WF = 137078.
[<کِتاب, k(i)tab>
14. {کِتاب, k(i)tab} (1074) N - NF Sg Nom - Fem
15. {کِتاب, k(i)tab} (1074) N - NF Sg Obl - Fem
16. {کِتاب, k(i)tab} (1074) N - NF Sg Voc - Fem
]
Tagger Mode: Please be patient! It will take some time to build the data sturcture first time...


javax.swing.JFrame:- Tagger Mode
Analysis
Utilities
Settings
Info
C-Trie program
Tagger Mode
Synthesiser Mode
Inflection Mode
Exit


javax.swing.JFrame:- Urdu Transliterator
Analysis
Utilities
Settings
Info
Conversion from Urdu to Roman
Conversion from Roman to Urdu
Urdu Script
آج کی بات
Roman Transliteration
Aj ky bat
Unicode
\u0622\u062c\u0020\u06a9\u06cc\u0020\u0628\u0627\u062a
Show
Clear
Urdu Transliterator: Running...


javax.swing.JFrame:- Unicode Urdu Extraction
Analysis
Utilities
Settings
Info
Text Extractor
Web/Local Address:
http://www.urduweb.org/ur/
Brows
Extract data
Text in Urdu Script:
اردو ویب کیا ہے؟
اردو جنوبی ایشیا کی زبانوں میں سے ایک عام بولی جانے والی زبان ہے جس کے بولنے والے دس
موجودہ زمانے میں اردو کی ترقی کے لئے اسے انٹرنیٹ اور کمپیوٹر پر فروغ دینے کی اشد ضرورت ہ
اردو ویب کا مقصد اردو سمجھنے والوں کو ایک دوسرے کے قریب لانا ہے اور اردو کو انٹرنیٹ پر م
اردو سے محبت کرنے والوں کے لئے انٹرنیٹ پر ایک گوشہ فراہم کرنا
اردو میں نیا مواد لکھنا اور پرنٹ کی دنیا سے مواد کو آنلائن لانا
آزاد مصدر اردو سافٹو یر بنانا
و یب اور کمپیوٹر سافٹو یر کو اردو میں مقامیانا
ان موضوعات پر لوگوں کی مدد کے لئے مضامین کا ذخیرہ اکٹھا کرنا
ان مقاصد کے پیش نظر ہم نے اب تک جو کام کیا ہے اس میں سے کچھ دائیں طرف کے روابط م
Text in Roman trasliteration:
ardwwyb kya hE?
ardw jnwby ayXya ky zbanwN myN sE ayk eam bwly janE waly zban hE js kE bwlnE
mwjwdh zmanE myN ardw ky trqy kE ly^E asE anTrnyT awr kmpywTr pr frwG dynE k
ardwwyb ka mqSd ardw smjjhnE walwN kw ayk dwsrE kE qryb lana hE awr ardw kw
ardw sE mHbt krnE walwN kE ly^E anTrnyT pr ayk gwXh frahm krna
ardw myN nya mwad lk|hna awr prnT ky dnya sE mwad kw Anlay^n lana
Azad mSdr ardw safTwyr bnana
wyb awr kmpywTr safTwyr kw ardw myN mqamyana
an mw|Zweat pr lwgwN ky mdd kE ly^E m|Zamyn ka ZKyrh akT|ha krna
an mqaSd kE pyX(i) n|zr hm nE ab tk jw kam kya hE as myN sE kc|h day^yN |trf kE rv
ardwwyb ayk r|Zakaranh say^T hE. hm cahtE hyN kh ayk r|Zakar kmywnTy bKkyl dyN j
agr Ap apny Azad mSdr safTwyr kw ardw myN mqamyanE kE KwahaN hyN tw hman
Text in Unicode
4\u0627\u0634\u0020\u0020\u0020\u0020\u0020\u0020\u0020\u0020\u0020\u0020\u0020\u0020\u0020\u0020\u0020\u000a\u0020\u000a\u000a\u0020\u0020\u0020\u0020\u0020
Save Urdu Script in html format
Save Urdu Script in text format
Save Roman Transliteration in text format
Clear
Unicode Urdu Extraction: Running...

Thanks for your attention

Questions / Comments

Homepage of the project

http://www.lama.univ-savoie.fr/~humayoun/UrduMorph/