Comput Syst Sci & Eng (2019) 3: 113–121

© 2019 CRL Publishing Ltd

International Journal of
Computer Systems
Science & Engineering

A Two-Level Morphological Description of Bashkir Turkish

Can Eyupoglu

Department of Computer Engineering, Istanbul Commerce University, Istanbul, 34840 Turkey
E-mail: ceyupoglu@ticaret.edu.tr

In recent years, the topic of Natural Language Processing (NLP) has attracted increasing interest. Many NLP applications including machine translation, machine learning, speech recognition, sentiment analysis, semantic search and natural language generation have been developed for most of the existing languages. Besides, two-level morphological description of the language to be used is required for these applications. However, there is no comprehensive study of Bashkir Turkish in the literature. In this paper, a two-level description of Bashkir Turkish morphology is described. The description based on a root word lexicon of Bashkir Turkish is implemented using Extensible Markup Language (XML) and appended to Nuve framework. The phonetic rules of Bashkir Turkish are encoded using 41 two-level rules. This two-level morphological description is promising to be used in Bashkir Turkish oriented NLP applications.

Keywords: Bashkir Turkish, Extensible Markup Language, Natural Language Processing, two-level morphology

1. INTRODUCTION

Bashkir, the co-official language with Russian in the Republic of Bashkortostan, is the part of the Kipchak group of the Turkic languages. There are almost 1.2 million people speaking Bashkir in the Russian Federation, with the ethnic population nearly 1.6 million according to the 2010 census data. Bashkir language has three dialects, namely Burzhan (Western Bashkir), Kuvakan (Mountain Bashkir) and Yurmaty (Steppe Bashkir) [1].

Bashkir is an agglutinative subject-object-verb language as a member of the Turkic language family [2]. In Bashkir, the vocabulary mostly consists of Turkic roots. Furthermore, Bashkir has lots of loan words from Arabic, Russian and Persian languages [3, 4].

In earlier times, Chagatai was used as the written language by Bashkir people and then replaced with a literary Turkic language which is a regional diversity of Turki in the late 19^{th} century. Turki and Chagatai were written in a variance of the Arabic script. A writing system for Bashkir was particularly created using the Arabic script in 1923. Concurrently, a literary Bashkir language using a modified Arabic alphabet in the beginning was formed by differing from Turkic influences. This Arabic alphabet was replaced with a Latin alphabet in 1930 and Cyrillic alphabet in 1938, respectively [4].

Bashkir Turkish is a bridge between Tatar and Kazakh Turkish, and has almost the same features with Tatar Turkish in terms of structure. Nevertheless, it moves away from Tatar Turkish in the way of phonology. Bashkir Turkish differs from historical written Turkic language with its distinctive lisp and fricative consonants. Besides, the advanced consonant harmonies are seen in Bashkir Turkish as in Kazak Turkish [5-11].

Bashkir Turkish has finite-state and highly complicated morphotactics as in Turkish language [12]. The words in Bashkir can be converted from a nominal structure to verbal structure or vice-versa by means of adding morphemes to a root word or a stem. These morphemes can also create adverb structures. The phonetic rules in Bashkir Turkish constrain and alter morphological structures. In order to achieve vowel harmony, vowels in affixed morphemes have to comply with the preceding vowel in definite circumstances. Moreover, vowels in the roots and morphemes are dropped under certain conditions. In a similar way, consonants in the roots or in the affixed morphemes experience certain modifications and might be removed.

Natural Language Processing (NLP) is the area of computational modelling of several aspects of natural languages and developing numerous systems [13]. In order to make computer systems discover and process languages, many NLP methods and applications have been developed in the disciplines of computer engineering, information science, linguistics and psychology. Machine learning, artificial intelligence, natural language generation, expert systems, speech recognition, machine translation, summarization, sentiment analysis and semantic search are the examples of NLP applications [14, 15]. Various studies including the aforementioned applications have been done for two-level morphological descriptions of many languages until now. To the best of the author's knowledge, in the literature, there is no other comprehensive work related to Bashkir Turkish in this framework. This paper describes a two-level morphological description based on a root word lexicon of Bashkir Turkish. The implementation of this morphological description promising to be used in Bashkir Turkish NLP applications is performed utilizing Extensible Markup Language (XML) and added to Nuve which is a two-level parser/generator framework developed for agglutinative languages.

The rest of the paper is organized as follows. In Section 2, two-level morphology is explained. Section 3 introduces the two-level morphological description of Bashkir Turkish. In Section 4, the implementation of two-level rules is demonstrated. Finally, conclusions being under study are summarized in Section 5.

2. TWO-LEVEL MORPHOLOGY

Two-level morphology is a generic approach to describe morphology of word structures [16-19] and used for analysing the morphology of various languages [12, 20-35]. Two-level description consists of two levels, namely lexical and surface. The structure of the functional components of a word is represented by the lexical level. On the other hand, the standard orthographic realization of the word associated with the given lexical structure is represented by the surface level [12, 16, 26]. The rule types denoting the phonetic restrictions and modifications are demonstrated in Table 1. Left context (LC) and right context (RC) denote lexical and surface levels, respectively.

Context restriction, surface coercion, composite and exclusion rules shown in Table 1 are separately compiled into a Finite State Transducer (FST) which is a Finite State Machine (FSM) consisting of lexical and surface tapes. These FSTs control whether a lexical matches a surface correspondingly [36, 37]. The FST architecture for two-level morphology is demonstrated in Figure 1.

Appropriate morpheme sequences are designated by morphotactics which are encoded as FSMs. Moreover, these FSMs utilize lexicons for roots and suffixes, and changes for obtaining suffix sequences [12]. Readers are referred to [16] for further details about two-level morphology.

3. TWO-LEVEL MORPHOLOGICAL DESCRIPTION

The Bashkir Turkish language is officially written in Cyrillic alphabet and its orthography is composed of an adapted alphabet of 35 Latin letters. There are 9 vowels: *a*, *ä*, *ı*, *i*, *η*, *u*, *ü*, *ŭ*, *u*, and 26 consonants: *b*, *v*, *d*, *g*, *ġ*, *η*, *j*, *z*, *y*, *k*, *q*, *l*, *m*, *n*, *η*, *p*, *r*, *s*, *š*, *t*, *f*, *h*, *ç*, *ş*, *x*, *w* [38]. In addition, there are geminate consonants, such as "*ts*" and "*şç*" taken from Russian. There are also "*yu*", "*yo*" and "*ya*" voices used in the Russian words. The phonetic features corresponding to the sounds denoted by these vowels and consonants are shown in Tables 2 and 3.

In order to create the two-level description of Bashkir Turkish morphology, firstly, the following letter subsets are defined:

1. Consonants: C = {b, v, d, g, ġ, z̧, j, z, y, k, q, l, m, n, *η*, p, r, s, š, t, f, h, ç, ş, x, w}
2. Lexical vowels: V = {a, ä, ı, i, í̆, u, ü, ŭ, ǖ̆}
3. Back vowels: V_b = {a, ı, u, ŭ}
4. Front vowels: V_f= {ä, í̆, i, ü, ǖ̆}
5. Front unrounded vowels: V_{fu} = {ä, ắ, i}
6. Front rounded vowels: V_{fr} = {ü, ǖ̆}
7. Back unrounded vowels: V_{bu} = {a, ı}
8. Back rounded vowels: V_{br} = {u, ŭ}
9. Lexical voiced consonants: C_{v+} = {b, d, g, ġ}
10. Lexical voiceless consonants: C_{v-} = {p, t, ts, ç, şç, k, q, h}
11. Lexical consonants used in some affixes and suffixes: L = {l, d, t, z̧}
12. Lexical unrounded low vowels: A = {a, ä}
13. Lexical consonants used in some affixes and suffixes: N = {n, d, t, z̧}
14. Lexical vowels used in some affixes and suffixes: H = {ı, í̆, ŭ, ǖ̆}
15. Lexical consonants used in some affixes and suffixes: G = {g, ġ, k, q}
16. Lexical vowels: V_I = {ä, í̆, i, ü}
17. Lexical vowels: V_K = {a, u, ı}

3.1 Two-Level Rules

The two-level rules for the phonetic component of the morphological description are given below:

1. L:l <= V +:0__Ar
This rule converts **L** which is at the beginning of the suffix +LAr to **l** when the last letter of stem is V.
Lexical: äsä+LAr N(mother/anne)+PLU
Surface: äsä0lär äsälär (mothers/anneler)

2. L:d <= [l | m | n | *η*] +:0__Ar
This rule converts **L** which is at the beginning of the suffix +LAr to **d** when the last letter of stem is one of the consonants in the option list.
Lexical: awıl+LAr N(village/köy)+PLU
Surface: awıl0dar awıldar (villages/köyler)

Table 1 The rule types of phonetic restrictions and modifications.

Rule type	Rule	Description
Context restriction	**a:b => LC __ RC**	**a** is realized as **b** only in the given **LC** and **RC**, but not necessarily always.
Surface coercion	**a:b <= LC __ RC**	**a** is always realized as **b** in the given **LC** and **RC**, but not necessarily only in this context.
Composite	**a:b <=> LC __ RC**	**a** is always realized as **b** in the given **LC** and **RC** and nowhere else.
Exclusion	**a:b / <= LC __ RC**	**a** is never realized as **b** in the given **LC** and **RC**.

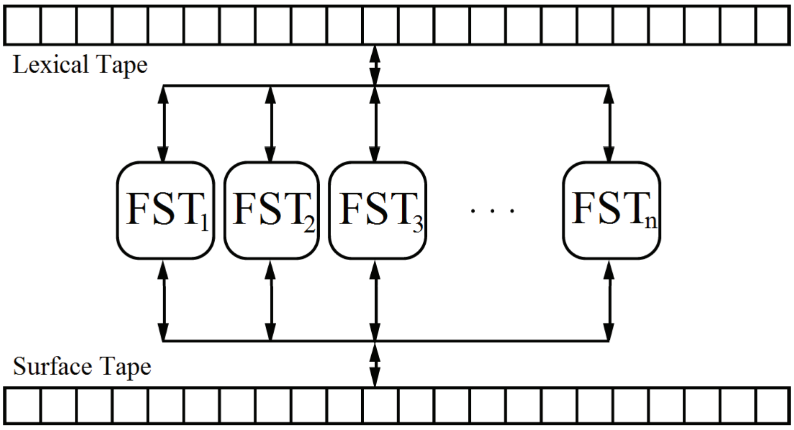


Figure 1 FST architecture for two-level morphology [12].

Table 2 Phonetic features of Bashkir Turkish vowels [38].

Vowels		Unrounded				Rounded	
		Low	Semi High	High		High	Semi High
Back		a		ı		u	ŭ
Front		ä	í́	i		ü	ü̆

Table 3 Phonetic features of Bashkir Turkish consonants [38].

Consonants		Labial	Labio- dental	Dental	Alveolar	Palato- alveolar	Velar	Uvular	Palato
Continuant	Nasal	m			n			η	
	Liquid						l, y		
	Trill						r		
	Fricative	w	f, v	š, z̧	s, z	j, ş		x	
Tenuis	Voiced	b			d		g	ġ	
	Voiceless	p			t, ts	ç, şç	k	q	h

3. L:t <= C_{v-} +:0__Ar
This rule converts **L** which is at the beginning of the suffix +LAr to **t** when the last letter of stem is C_{v-}.
Lexical: aġas+LAr N(tree/ağaç)+PLU
Surface: aġas0tar aġastar (trees/ağaçlar)

4. L:η <= [r | y | z̧ | w] +:0__Ar
This rule converts **L** which is at the beginning of the suffix +LAr to η when the last letter of stem is one of the consonants in the option list.
Lexical: hıyır+LAr N(cow/inek)+PLU
Surface: hıyır0z̧ar hıyırz̧ar (cows/inekler)

5. N:n <= V +:0__Hη
This rule converts **N** which is at the beginning of the suffix +NHη to **n** when the last letter of stem is V.
Lexical: alma+NHη N(apple/elma)+GEN
Surface: alma0nıη almanıη (… of apple/elmanın)

6. N:d <= [l | m | n | η] +:0__Hη
This rule converts **N** which is at the beginning of the suffix +NHη to **d** when the last letter of stem is one of the consonants in the option list.
Lexical: ŭn+NHη N(flour/un)+GEN
Surface: ŭn0dŭη ŭndŭη (… of flour/unun)

7. N:t <= C_{v-} +:0__Hη
This rule converts **N** which is at the beginning of the suffix +NHη to **t** when the last letter of stem is C_{v-}.
Lexical: qunaq+NHη N(guest/misafir)+GEN
Surface: qunaq0tıη qunaqtıη (… of guest/misafirin)

8. N:η <= [r | y | η | w] +:0__Hη
This rule converts **N** which is at the beginning of the suffix +NHη to **z̧** when the last letter of stem is one of the consonants in the option list.
Lexical: ǚy+NHη N(house/ev)+GEN
Surface: ǚy0z̧ǚη ǚyz̧ǚη (… of house/evin)

9. N:n <= V +:0__H
This rule converts **N** which is at the beginning of the suffix +NH to **n** when the last letter of stem is V.
Lexical: baqsa+NH N(garden/bahçe)+ACC
Surface: baqsa0nı baqsanı (the garden/bahçeyi)

10. N:d <= [l | m | n | η] +:0__H
This rule converts **N** which is at the beginning of the suffix +NH to **d** when the last letter of stem is one of the consonants in the option list.
Lexical: urman+NH N(forest/orman)+ACC
Surface: urman0dı urmandı (the forest/ormanı)

11. N:t <= C_{v-} +:0__H
This rule converts **N** which is at the beginning of the suffix +NH to **t** when the last letter of stem is C_{v-}.
Lexical: kitap+NH N(book/kitap)+ACC
Surface: kitap0tı kitaptı (the book/kitabı)

12. N:z̧ <= [r | y | z̧ | w] +:0__H
This rule converts **N** which is at the beginning of the suffix +NH to **z̧** when the last letter of stem is one of the consonants in the option list.
Lexical: küz̧+NH N(eye/göz)+ACC
Surface: küz̧0z̧ǚ küz̧z̧ǚ (the eye/gözü)

13. L:l <= V +:0__A
This rule converts **L** which is at the beginning of the suffix +LA to **l** when the last letter of stem is V.
Lexical: tantana+LA N(ceremony/tören)+LOC
Surface: tantana0la tantanala (at ceremony/törende)

14. L:d <= [l | m | n | η] +:0__A
This rule converts **L** which is at the beginning of the suffix +LA to **d** when the last letter of stem is one of the consonants in the option list.
Lexical: qul+LA N(hand/el)+LOC
Surface: qul0da qulda (on hand/elde)

15. L:t <= C_{v-} +:0__A
This rule converts **L** which is at the beginning of the suffix +LA to **t** when the last letter of stem is C_{v-}.
Lexical: tŭrmŭş+LA N(life/hayat)+LOC
Surface: tŭrmŭş0ta tŭrmŭşta (in life/hayatta)

16. L:z̧ <= [r | y | z̧ | w] +:0__A
This rule converts **L** which is at the beginning of the suffix +LA to η when the last letter of stem is one of the consonants in the option list.
Lexical: yäy+LA N(summer/yaz)+LOC
Surface: yäy0z̧ä yäz̧ηä (in summer/yazda)

17. N:n <= V +:0__An
This rule converts **N** which is at the beginning of the suffix +NAn to **n** when the last letter of stem is V.
Lexical: bisä+NAn N(woman/kadın)+ABL
Surface: bisä0nän bisänän (from woman/kadından)

18. N:d <= [l | m | n | η] +:0__An
This rule converts **N** which is at the beginning of the suffix +NAn to **d** when the last letter of stem is one of the consonants in the option list.
Lexical: mŭrŭn+NAn N(nose/burun)+ABL
Surface: mŭrŭn0dan mŭrŭndan (from nose/burundan)

19. N:t <= C_{v-} +:0__An
This rule converts **N** which is at the beginning of the suffix +NAn to **t** when the last letter of stem is C_{v-}.
Lexical: bílgís+NAn N(expert/uzman)+ABL
Surface: bílgís0tän bílgístän (from expert/uzmandan)

20. N:z̧ <= [r | y | z̧ | w] +:0__An
This rule converts **N** which is at the beginning of the suffix +NAn to **z̧** when the last letter of stem is one of the consonants in the option list.
Lexical: bísäy+NAn N(cat/kedi)+ABL
Surface: bísäy0z̧än bísäyz̧än (from cat/kediden)

21. G:g <=> [V | C_{v+}] +:0__ä
This rule converts **G** which is at the beginning of the suffix +Gä to **g** when the last letter of stem is one of V or C_{v+}.
Lexical: güzäl+Gä N(beautiful/güzel)+DAT
Surface: güzäl0gä güzälgä (güzele)

22. G:ġ <=> [V | C_{v+}] +:0__a
This rule converts **G** which is at the beginning of the suffix +Ga to **ġ** when the last letter of stem is one of V or C_{v+}.
Lexical: baz̧ar+Ga N(bazaar/çarşı)+DAT
Surface: baz̧ar0ġa baz̧arġa (to bazaar/çarşıya)

23. G:k <=> C_{v-} +:0__ä
This rule converts **G** which is at the beginning of the suffix +Gä to **k** when the last letter of stem is C_{v-}.
Lexical: biş+Gä N(five/beş)+DAT
Surface: biş0kä bişkä (beşe)

24. G:q <=> C_{v-} +:0__a
This rule converts **G** which is at the beginning of the suffix +Ga to **q** when the last letter of stem is C_{v-}.
Lexical: maqsat+Ga N(purpose/amaç)+DAT
Surface: maqsat0qa maqsatqa (amaca)

25. L:l <= V +:0__Ay
This rule converts **L** which is at the beginning of the suffix +LAy to **l** when the last letter of stem is V.
Lexical: bala+LAy N(child/çocuk)+SIM
Surface: bala0lay balalay (like child/çocuk gibi)

26. L:d <= [l | m | n | η] +:0__Ay
This rule converts **L** which is at the beginning of the suffix +LAy to **d** when the last letter of stem is one of the consonants in the option list.
Lexical: säsän+LAy N(bard/ozan)+SIM
Surface: säsän0däy säsändäy (like bard/ozan gibi)

27. L:t <= C_{v-} +:0__Ay

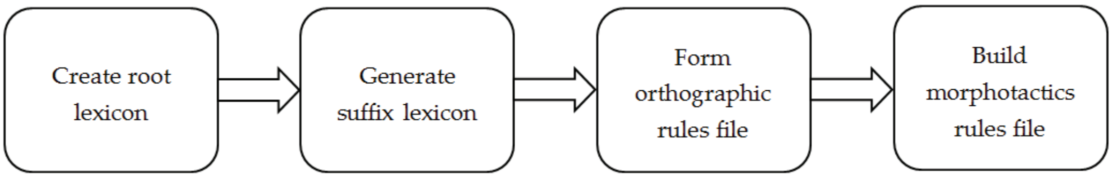
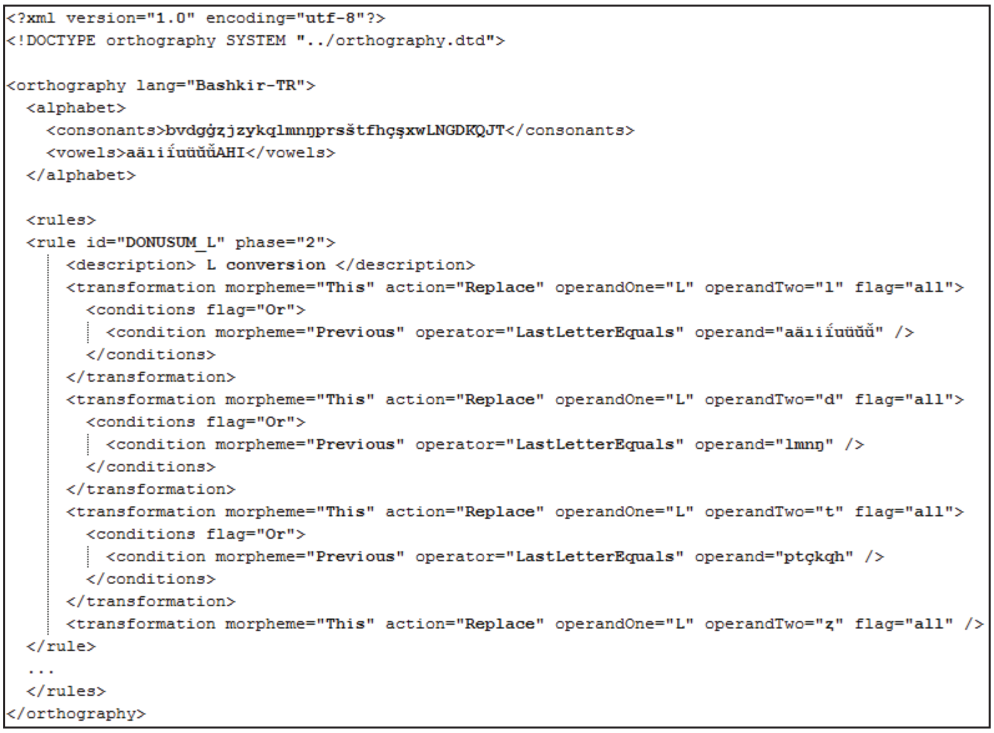


Figure 2 Implementation outline of two-level morphological description.



```
<?xml version="1.0" encoding="utf-8"?>
<!DOCTYPE orthography SYSTEM "../orthography.dtd">

<orthography lang="Bashkir-TR">
  <alphabet>
    <consonants>bvdgġẓjzykqlmnŋprsštfhçşxwLNGDKQJT</consonants>
    <vowels>aäıiḯuüŭ̈ǚAHI</vowels>
  </alphabet>

  <rules>
  <rule id="DONUSUM_L" phase="2">
      <description> L conversion </description>
      <transformation morpheme="This" action="Replace" operandOne="L" operandTwo="l" flag="all">
        <conditions flag="Or">
          <condition morpheme="Previous" operator="LastLetterEquals" operand="aäıiḯuüŭ̈ǚ" />
        </conditions>
      </transformation>
      <transformation morpheme="This" action="Replace" operandOne="L" operandTwo="d" flag="all">
        <conditions flag="Or">
          <condition morpheme="Previous" operator="LastLetterEquals" operand="lmnŋ" />
        </conditions>
      </transformation>
      <transformation morpheme="This" action="Replace" operandOne="L" operandTwo="t" flag="all">
        <conditions flag="Or">
          <condition morpheme="Previous" operator="LastLetterEquals" operand="ptçkqh" />
        </conditions>
      </transformation>
      <transformation morpheme="This" action="Replace" operandOne="L" operandTwo="ẓ" flag="all" />
  </rule>
  ...
  </rules>
</orthography>
```

Figure 3 A part of the orthographic rules file.

This rule converts **L** which is at the beginning of the suffix +LAy to **t** when the last letter of stem is C_{v-}.
Lexical: qŭ̈ş+LAy N(bird/kuş)+SIM
Surface: qŭ̈ş0tay qŭ̈ştay (like bird/kuş gibi)

28. L:ẓ <= [r | y | ẓ | w] +:0__Ay
This rule converts **L** which is at the beginning of the suffix +LAy to **ẓ** when the last letter of stem is one of the consonants in the option list.
Lexical: taw+LAy N(mountain/dağ)+SIM
Surface: taw0ẓay tawẓay (like mountain/dağ gibi)

29. k:g <=> __+V
This rule converts the consonant **k** which is at the end of a stem to **g** when a suffix starting with a vowel is affixed.
Lexical: kǚrḯk+Hm N(shovel/kürek)+Poss1PS
Surface: kǚrḯg0ḯm kǚrḯgḯm (my shovel/küreğim)

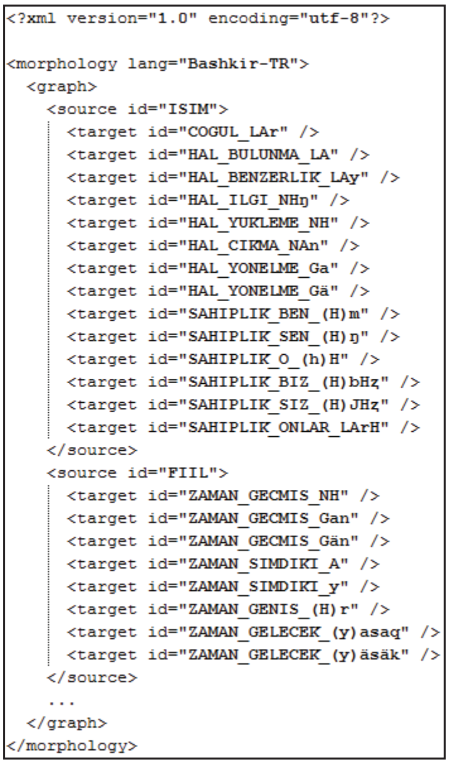
30. q:ġ <=> __+V
This rule converts the consonant **q** which is at the end of a stem to **ġ** when a suffix starting with a vowel is affixed.
Lexical: ayaq+H N(foot/ayak)+Poss3PS
Surface: ayaġ0ı ayaġı (his foot/ayağı)

31. p:b <=> __+V
This rule converts the consonant **p** which is at the end of a stem to **b** when a suffix starting with a vowel is affixed.
Lexical: qap+Hm N(container/kap)+Poss1PS
Surface: qab0ım qabım (my container/kabım)

32. 0:n <=> (h)H__+ L:@ A:@
This rule deals with the case when a new consonant is added on the surface. The word ending with 3^{rd} person single possessive suffix gets a **n** consonant between the locative and possessive suffixes.
Lexical: bändähḯ+LA N(his slave/onun kölesi)+Poss3PS+LOC
Surface: bändähḯ0ndä bändähḯndä (in his slave/onun kölesinde)

33. 0:H <=> C__+C:@
If the word ending with consonant or semi-consonant and to affix possessive 1^{st}, 2^{nd} person single and plural suffix gets one of the helper vowel **H** between the word and morpheme on the surface.
Lexical: ḯş+Hm N(flower/çiçek)+Poss1PS
Surface: ḯş0ḯm ηşḯm (my flower/çiçeğim)

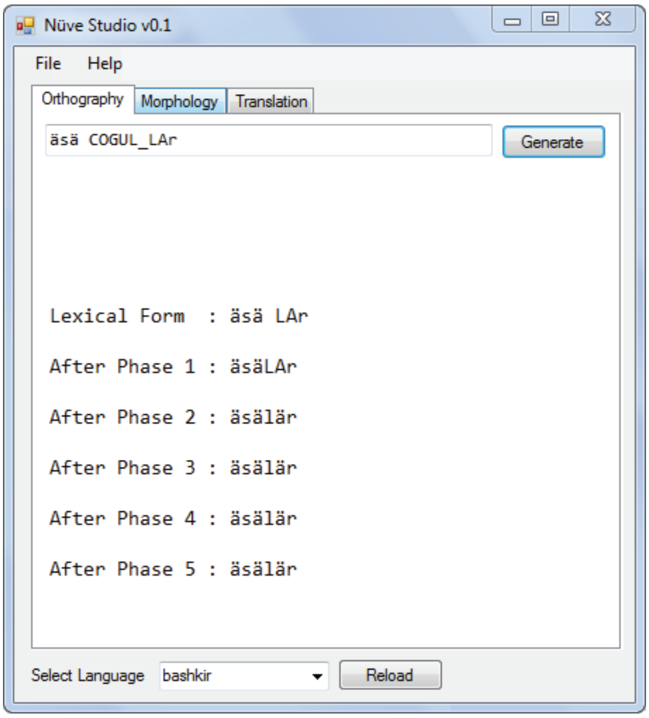
34. V:0 => V+:0__
If both ending letter of the word and beginning letter of the suffix are vowels then the first letter of suffix is removed.
Lexical: bala+Hη N(child/çocuk)+Poss2PS
Surface: bala0η balaη (child's/çocuğun)



```
<?xml version="1.0" encoding="utf-8"?>

<morphology lang="Bashkir-TR">
  <graph>
    <source id="ISIM">
      <target id="COGUL_LAr" />
      <target id="HAL_BULUNMA_LA" />
      <target id="HAL_BENZERLIK_LAy" />
      <target id="HAL_ILGI_NHŋ" />
      <target id="HAL_YUKLEME_NH" />
      <target id="HAL_CIKMA_NAn" />
      <target id="HAL_YONELME_Ga" />
      <target id="HAL_YONELME_Gä" />
      <target id="SAHIPLIK_BEN_(H)m" />
      <target id="SAHIPLIK_SEN_(H)ŋ" />
      <target id="SAHIPLIK_O_(h)H" />
      <target id="SAHIPLIK_BIZ_(H)bHȥ" />
      <target id="SAHIPLIK_SIZ_(H)JHȥ" />
      <target id="SAHIPLIK_ONLAR_LArH" />
    </source>
    <source id="FIIL">
      <target id="ZAMAN_GECMIS_NH" />
      <target id="ZAMAN_GECMIS_Gan" />
      <target id="ZAMAN_GECMIS_Gän" />
      <target id="ZAMAN_SIMDIKI_A" />
      <target id="ZAMAN_SIMDIKI_y" />
      <target id="ZAMAN_GENIS_(H)r" />
      <target id="ZAMAN_GELECEK_(y)asaq" />
      <target id="ZAMAN_GELECEK_(y)äsäk" />
    </source>
    ...
  </graph>
</morphology>
```

Figure 4 A part of the morphotactics rules file.



```
äsä COGUL_LAr

Lexical Form  : äsä LAr

After Phase 1 : äsäLAr

After Phase 2 : äsälär

After Phase 3 : äsälär

After Phase 4 : äsälär

After Phase 5 : äsälär
```

Figure 5 User interface of morphological generation for Bashkir Turkish in Nuve.

35. H:0 <=> V+C__C+(C)V

H can state at closed second syllables when the suffix how there is vowel in is affixed to word and the vowel is deleted.

Lexical: uyın+V N(play/oynamak)+NtoV
Surface: uy0na uyna (play/oynamak)

36. V:i̇́ => V_f(C) + (C)__

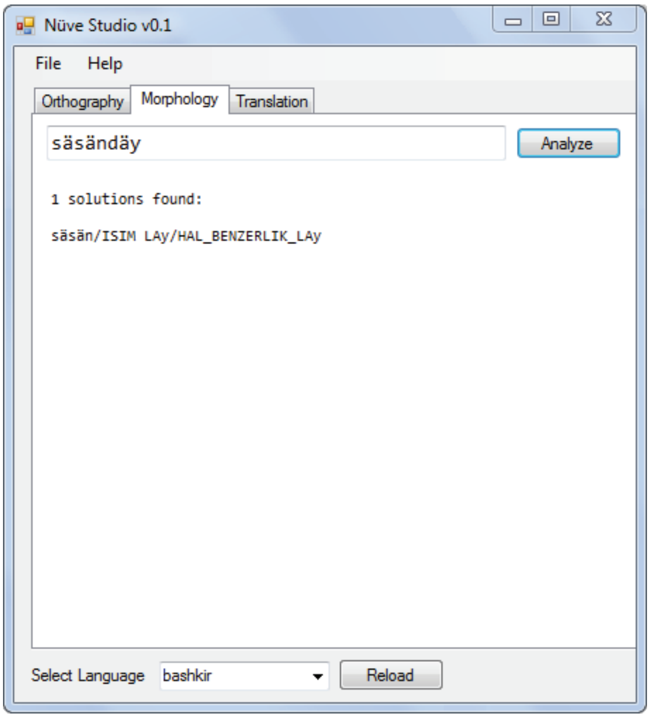
If a vowel in a syllable is V_f, the vowel at the suffix will be **i̇́**.

Lexical: ikmäk+tVη N(bread/ekmek)+GEN
Surface: ikmäk0ti̇́η ikmäkti̇́η (… of bread/ekmeğin)

37. V:a => V_b(C) + (C)__

If a vowel in a syllable is V_b, the vowel at the suffix will be **a**.

Lexical: qŭş+tVy N(bird/kuş)+SIM



säsändäy

```
1 solutions found:

säsän/ISIM LAy/HAL_BENZERLIK_LAy
```

Figure 6 User interface of morphological parsing for Bashkir Turkish in Nuve.

Surface: qŭş0tay qŭştay (like bird/kuş gibi)

38. V:ä => V_I(C) + (C)__
If a vowel in a syllable is V_I, the vowel at the suffix will be **ä**.
Lexical: imän+dVy N(billet/kütük)+SIM
Surface: imän0däy imändäy (like billet/kütük gibi)

39. V:ı => V_K(C) + (C)__
If a vowel in a syllable is V_K, the vowel at the suffix will be **ı**.
Lexical: kural+hVz̧ N(weapon/silah)
Surface: kural0hız̧ kuralhız̧ (unarmed/silahsız)

40. V:ǚ => ǚ + C + (C)__
If a vowel in a syllable is ǚ, the vowel at the suffix will be ˇ¨**u**.
Lexical: kǚz̧gǚ+nVη N(mirror/ayna)+GEN
Surface: kǚz̧gǚ0nǚη kǚz̧gǚnǚη (… of mirror/aynanın)

41. V:ŭ => ŭ+C + (C)__
If a vowel in a syllable is ŭ, the vowel at the suffix will be **ŭ**.
Lexical: ŭn+dVη N(flour/un)+POSS
Surface: ŭn0dŭη ŭndŭη (… of flour/unun)

4. IMPLEMENTATION OF TWO-LEVEL RULES

In this study, a two-level morphological description including 41 two-level rules generated for the phonetic rules of Bashkir Turkish is described. The description is implemented using XML and added to Nuve framework. A lexicon of approximately 700 words is created and utilized for implementation and testing. After implementation, all words in the lexicon have been tested and it has been observed that morphological generation and parsing function well for all words, which means a test accuracy of one hundred percent.

Nuve [39] is a language independent top-down morphological analyser and generator designed principally for Turkic languages, and can be utilized for all agglutinative languages. It is open source and developed with C# on .NET platform. Nuve also supports stemming, sentence boundary detection and n-gram extraction.

The implementation outline of the two-level description of Bashkir Turkish morphology consisting of the following four steps is shown in Figure 2.

Step 1: A root lexicon for Bashkir Turkish containing root type and flag attributes is created as a comma-separated values (CSV) file.

Step 2: A suffix lexicon for Bashkir Turkish including lexical form, surface form and rule type attributes is generated as a CSV file.

Step 3: An orthographic rules file involving the two-level rules for the phonetic component of the morphological description is formed in XML format. A part of the orthographic rules file is shown in Figure 3 which contains Bashkir Turkish alphabet and indicates the 1^{st}, 2^{nd}, 3^{rd} and 4^{th} two level rules, respectively.

Step 4: A morphotactics rules file is created for Bashkir Turkish in XML format. Figure 4 demonstrates a part of the morphotactics rules file.

After all language specific files, such as root lexicon, suffix lexicon, orthographic and morphotactics rules are defined, morphological generation and parsing for Bashir Turkish can be tested on Nuve. The user interfaces of Nuve for orthography and morphology are shown in Figures 5 and 6, respectively.

In the morphological generation stage, a desired root/stem is designated with one or more suffixes and then Nuve generates the surface forms as 5 phases according to the lexical form of the root/stem specified in suffix lexicon file (Figure 5).

In other respects, morphological parsing of a chosen Bashkir Turkish word is realised by Nuve as shown in Figure 6.

5. CONCLUSION

In this paper, a two-level morphological description based on a root word lexicon is described for Bashkir Turkish. This description is implemented using XML and added to Nuve framework that is an agglutinative language independent two-level generator/parser developed especially for Turkic languages. The phonetic rules of Bashkir Turkish are encoded utilizing 41 two-level rules. Furthermore, a root lexicon of about 700 words is used for implementation and testing stages. Being the first extensive two-level description of Bashkir Turkish, this two-level morphological description is promising to be used to feed Bashkir Turkish-based NLP applications, such as corpus tagging, text segmentation and semantic analysis.

REFERENCES

1. Ethnologue: Languages of the World, Bashkort: A language of Russian Federation, 2018, https://www.ethnologue.com/language/bak
2. E. Yavuz and V. Topuz, “A phoneme-based approach for eliminating out-of-vocabulary problem of Turkish speech recognition using Hidden Markov Model,” *Computer Systems Science and Engineering*, Vol. 33, No. 6, 2018, pp. 429-445.
3. L. Johanson and É. Á. Csató, (eds.), *The Turkic languages: Tatar and Bashkir*, New York, NY: Routledge, 1998.
4. Overview of the Bashkir Language, Learn the Bashkir Language & Culture, 2011, http://www.transparent.com/learn-bashkir/overview. html
5. J. Benzing, “Das Baschkirische,” *PhTF I*, 1959, pp. 421-434 (M. Argunşah, “Başkurt Türkçesi,” *Türk Dünyası Araştırmaları*, Vol. 1, 1995, pp. 127-142).
6. N. K. Dmitriyev, *Grammatika başkirskogo yazıka*, Moskova-Leningrad, 1948.
7. N. K. Dmitriyev, *Başqŭrt Tílíníη Grammatikahı*, Ufa, 1950.
8. N. H. İşbulatov, *XäÂírgí Başqŭrt Tílí (HüÂ türkümdäriniη bülüníşí)*, Ufa, 1972.
9. A. A. Yuldaşev, *Başkirskiy Yazık, Yaziki Mira: Tyurksie Yazıki*, (red. E. R. Tenişev), Bişkek, 1997, pp. 206-216.
10. M. Öner, *Bugünkü Kıpçak Türkçesi*, Ankara, TDK, 1998.
11. H. Y. Ersoy, “Bashkir Turks and Their Language,” *Journal of Endangered Languages: Turkic Languages*, Vol. 3, No. 4-5, 2014, pp. 147-191.
12. K. Oflazer, “Two-level description of Turkish morphology,” *Literary and Linguistic Computing*, Vol. 9, No. 2, 1994, pp. 137-148.
13. A. K. Joshi, “Natural Language Processing,” *Science*, Vol. 253, 1991, pp. 1242-1249.
14. G. G. Chowdhury, “Natural language processing,” *Annual Review of Information Science and Technology*, Vol. 37, 2003, pp. 51-89.
15. S. K. Metin, “Neighbour unpredictability measure in multiword expression extraction,” *Computer Systems Science and Engineering*, Vol. 31, No. 3, 2016, pp. 209-221.
16. R. Sproat, *Morphology and Computation*, MIT Press, 1992.
17. K. Koskenniemi, “Two-level morphology: A general computational model for word form recognition and production,” Publication No: 11, Department of General Linguistics, University of Helsinki, 1983.
18. L. Karttunen, “KIMMO: a general morphological processor,” in *Proceedings of Texas Linguistic Forum*, Vol. 22, 1983, pp. 163-186.
19. E. L. Antworth, “PC-KIMMO: A two-level processor for Morphological Analysis,” Summer Institute of Linguistics, Dallas, Texas, 1990.
20. K. Koskenniemi, “An application of the two-level model to Finnish,” Computational Morphosyntax: a report on reseach 1981-1984, University of Helsinki, Department of General Linguistics, 1985.
21. S. Lun, “A two-level morphological analysis of French,” in *Proceedings of Texas Linguistic Forum*, Vol. 22, 1983, pp. 271-278.
22. R. Khan, “A two-level morphological analysis of Rumanian,” in *Proceedings of Texas Linguistic Forum*, Vol. 22, 1983, pp. 253-270.
23. L. Karttunen and K. Wittenburg, “A two-level morphological analysis of English,” in *Proceedings of Texas Linguistic Forum*, Vol. 22, 1983, pp. 217-228.
24. Y. S. Alam, “A two-level morphological analysis of Japanese,” in *Proceedings of Texas Linguistic Forum*, Vol. 22, 1983, pp. 229-252.
25. D. B. Kim, S. J. Lee, K. S. Choi, and G. C. Kim, “A two-level morphological analysis of Korean,” in *Proceedings of the 15th conference on Computational linguistics*, Kyoto, Japan, 5-9 August 1994, pp. 535-539.
26. A. C. Tantuğ, E. Adalı, and K. Oflazer, “Computer analysis of the Turkmen language morphology,” *Advances in Natural Language Processing*, Springer, Berlin, Heidelberg, 2006, pp. 186-193.
27. T. Y. Jang, “A two-level morphological analysis of Korean,” in *Proceedings of the Postgraduage Conference*, Department of Linguistics and Applied Linguistics, The University of Edinburgh, 1998.
28. H. R. Zafer, B. Tilki, A. Kurt, and M. Kara, “Two-Level Description of Kazakh Morphology,” in *Proceedings of the 1st International Conference on Foreign Language Teaching and Applied Linguistics*, Sarajevo, 5-7 May 2011, pp. 560-564.
29. E. Gökgöz, A. Kurt, K. Kulamshaev, and M. Kara, “Two-Level Qazan Tatar Morphology,” in *Proceedings of the 1st International Conference on Foreign Language Teaching and Applied Linguistics*, Sarajevo, 5-7 May 2011, pp. 428-432.
30. E. Gökgöz, K. Kulamshaev, H. R. Zafer, S. Öztoprak, İ. Biner, and A. Kurt, “An Implementation of Tatar Orthography Using The Nüve Framework,” in *Proceedings of the 3rd International Conference on Computer Processing In Turkic Languages*, Kazan, Russia, 8-10 April 2015.
31. K. Altintaş and İ. Çicekli, “A Morphological Analyser for Crimean Tatar,” in *Procedings of the 10th Turkish Symposium on Artificial Intelligence and Neural Networks*, North Cyprus, 2011, pp. 180-189.
32. Ç. Çöltekin, “A Freely Available Morphological Analyzer for Turkish,” in *Proceedings of the International Conference on Language Resources and Evaluation*, Valletta, Malta, 17-23 May 2010, pp. 820-827.
33. Z. Görmez, S. Ü. Baki, A. Kurt, K. Kulamshaev, and M. Kara, “An Overview of Two-Level Finite State Kyrgyz Morphology,” in *Proceedings of the 2nd International Symposium on Computing in Science & Engineering*, Aydın, Turkey, 2011.
34. Z. Yiner, A. Kurt, K. Kulamshaev, and H. R. Zafer, “Kyrgyz Orthography and Morphotactics with Implementation in NUVE” in *Proceedings of the International Conference on Engineering and Natural Science*, Sarajevo, 24-28 May 2016, pp. 1-8.
35. F. M. Tyers, J. N. Washington, I. Salimzyanov, and R. Batalov, “A prototype machine translation system for Tatar and Bashkir based on free/open-source components” in *Proceedings of the First Workshop on Language Resources and Technologies for Turkic Languages*, İstanbul, Turkey, 21 May 2012, pp. 11-14.

36. D. Jurafsky, *Speech and Language Processing*, Pearson, 2009.
37. M. Mohri, “Finite-state transducers in language and speech processing,” *Computational linguistics*, Vol. 23, No. 2, 1997, pp. 269-311.
38. H. Y. Ersoy, *Başkurt Türkçesinde Kip*, Doctoral dissertation, Gazi University, Ankara, Turkey, 2007.
39. H. R. Zafer, Nuve: A Natural Language Processing Library for Turkish in C#, 2018, https://github.com/hrzafer/nuve