Postupci i statistike slogovanja za talijanski jezik

Pokos, Marija

Undergraduate thesis / Završni rad

2016

Degree Grantor / Ustanova koja je dodijelila akademski / stručni stupanj: **University of Rijeka, Faculty of Humanities and Social Sciences / Sveučilište u Rijeci, Filozofski fakultet**

Permanent link / Trajna poveznica: https://urn.nsk.hr/urn:nbn:hr:186:643135

Rights / Prava: In copyright / Zaštićeno autorskim pravom.

Download date / Datum preuzimanja: **2024-11-17**

FILOZOFSKI FAKULTET U RIJECI

Sveučilište u Rijeci
University of Rijeka

Repository / Repozitorij:

Repository of the University of Rijeka, Faculty of Humanities and Social Sciences - FHSSRI Repository

zir.nsk.hr

UNIRI DIGITALNA KNJIŽNICA

dabar
DIGITALNI AKADEMSKI ARHIVI I REPOZITORIJI

Sveučilište u Rijeci – Odjel za informatiku

Preddiplomski dvopredmetni studij informatike i talijanskog jezika i književnosti

Marija Pokos

Postupci i statistike slogovanja za talijanski jezik

Završni rad

Mentor: izv. prof. dr. sc. Sanda Martinčić – Ipšić, dipl. ing.

Rijeka, kolovoz 2016.

ODJEL ZA INFORMATIKU
SVEUČILIŠTE U RIJECI
Radmile Matejčić 2, 51000 Rijeka, www.inf.uniri.hr

Rijeka, 11.3.2016.

Zadatak za završni rad

Pristupnica: **Marija Pokos**

Naziv završnog rada: Postupci i statistike slogovanja na talijanskom jeziku

Naziv završnog rada na eng. jeziku: Syllabification and statistics of the Italian language

Sadržaj zadatka: U završnome radu potrebno je proučiti principe i postupke za slogovanje u talijanskom jeziku. Rad će se primarno bazirati na postojećem leksikonu talijanskog jezika s rastavom na slogove ali će se predložiti i izraditi vlastiti postupci slogovanja koji će se usporediti s postojećim resursom. U završnome radu će se prikazati i statistika slogovanja talijanskog jezika.

Mentor

izv. prof. Sanda Martinčić-Ipšić

Voditelj za završne radove

dr. sc. Miran Pobar

Zadatak preuzet:

Pokos Marija

(potpis pristupnika)

Postupci i statistike slogovanja za talijanski jezik

SAŽETAK

Rastavljanje riječi na slogove, često nazivano i slogovanje, postupak je kojim se riječ rastavlja na slogove. Cilj ovog rada je izrada parsera i implementacija pravila za rastavljanje riječi na slogove u talijanskom jeziku za postojeći talijanski rječnik. Rječnik se sastoji od 440084 riječi od kojih je dio izvornih talijanskih, dio prisvojenih iz drugih jezika i sastavljen je od tri stupca. Najprije je potrebno urediti rječnik u dvije datoteke kako bi se u jednoj nalazila riječ i vrsta riječi, a u drugoj riječ i ta ista riječ rastavljena na slogove fonetskim znakovima jer je tako zadano u početnoj datoteci. Nakon toga trebalo je urediti treći stupac koji izgleda ovako: (((a1) 1) ((b a) 0) ((k i) 0))), a prvo parsiranje sastojalo se od micanja zagrada i nula, te dodavanja povlaka između slogova. Jedinice su ostale kako bi se znalo koji slog je naglašen. Nakon parsiranja dobio se niz kao ovaj: a11-ba-ki. Zatim je bilo potrebno implementirati pravila slogovanja koja će rastavljati riječi na slogove, ali u ortografskom zapisu. Primjećujemo da smo dobili dva različita zapisa riječi, jedan ortografski zapis i jedan fonetski zapis. Iz tog razloga potrebno je vratiti početni fonetski zapis u ortografski kako bi mogli usporediti početni zapis sa dobivenim. Vraćanje riječi u ortografski zapis potrebno je raditi za svako slovo posebno jer postoje jednaki fonetski znakovi koji imaju različito značenje za druga slova. Na primjer fonetski znak /dZi/ kod slova g se vraća u ortografsko [gi], dok se kod slova j vraća u ortografski [ji]. Potrebno je napomenuti da fonetski zapis koji se koristi u ovom rječniku nije u potpunosti izvoran nego je djelomično izmijenjen zbog lakšeg razumijevanja. Zbog razlike u zapisu dolazi do grešaka kod vraćanja u ortografski zapis, a do nekih grešaka i dolazi kod rastavljanje riječi pravilima. Analizom su utvrđene razlike između fonetskog i ortografskog zapisa, a postoci pogreške iznose: za fonetski i ortografski zapis: 77,29% pogreške; za ortografske zapise: 42,12% pogreške; nakon parsiranja ispravilo se 35,17% pogrešnih zapisa.

Ključne riječi: slog, slogovanje, ortografski zapis, fonetski zapis, pogreška, fonem

Syllabification and statistics of the Italian language

ABSTRACT

Disunion of words in syllables, often named syllabification is a process that is used for disunioning one word in syllables. The main objective of this research is a production of parser and implementation of the rules for disunioning words in syllables in Italian language for existing Italian dictionary. Dictionary consists of 440084 words, one part of original Italian words, the other part of words adopted from other languages and it consists of three columns. First it was needed to organize the dictionary in two folders in mode that one folder contains a word and its part of speech, and the other folder consists of a word and that same word divided in syllables with phonetics signs specified in the initial file. The next step was to reorganize the third column that look like this: (((a1) 1) ((b a) 0) ((k i) 0))), the first parsing moved brackets and zeros and added hyphen between the syllables. The ones remained because they marked the stress. After parsing a string that look like this was obtained: a11-ba-ki. After that it was required to implement the rules for syllabification that will divide words in syllables but in orthographic notation. It is noticed that we have two different notation of words, one orthographic and other phonetic notation. Because of that it is needed to reorganize the initial phonetic notation in orthographic notation so we can compare initial notation with the one we get. Backtracking of words in orthographic notation is required to do for each letter separately because there are phonetics signs that are equal for different letters. For example, phonetic sign /dZi/ for the letter g turns in orthographic [gi], and for letter j turns in orthographic [ji]. It is necessary to say that the phonetic notation that is used in this dictionary is not completely original, but is a little bit modified because of easier understanding. Because of the difference in notation, errors in backtracking from phonetic to orthographic notation appear and some errors occur during division of words implementing rules. Analysis show the difference between phonetic and orthographic notation and percentages of errors amount: for phonetic and orthographic notation: 77,29% of error; for orthographic notation: 42,12% of error; after parsing 35,17% of incorrect notation were corrected.

Keywords: syllable, syllabification, orthographic notation, phonetic notation, error, phoneme

Procedure e statistiche di sillabazione della lingua italiana

SOMMARIO

Divisione delle parole in sillabe, spesso chiamato anche sillabazione è un processo con il quale la parola viene divisa in sillabe. Lo scopo della ricerca è fare un analizzatore grammaticale e implementazione delle regole per sillabazione nella lingua italiana per un dizionario ricevuto. Il dizionario contiene 440084 parole, una parte delle parole italiane originali e l'altra parte delle parole straniere ed è fatto da tre colonne. Prima era necessario organizzare il dizionario in due file in modo che uno contiene la parola e il suo significato e l'altro la stessa parola divisa in sillabe con i segni fonetici come è nel dizionario iniziale. Dopo di questo era necessario riorganizzare la terza colonna di tipo: (((a1) 1) ((b a) 0) ((k i) 0))), in modo che si movessero le parentesi e i numeri zero e che si aggiungessero le trattini tra le sillabe. Il numero uno è rimasto perché mostra la sillaba accentata. Dopo l'analisi abbiamo la serie di tipo: a11-ba-ki. Il passo seguente era l'implementazione delle regole le quali dividerebbero la parola in sillabe però ortograficamente. Adesso ci sono due notazioni diverse, cioè uno ortografica e l'altra fonetica. A causa di questo dato è necessario tornare la notazione iniziale fonetica nella notazione ortografica in modo che potessimo comparare la notazione iniziale con quella eseguita. Il cambiamento dalla notazione ortografica alla fonetica è necessario fare per ognuna lettera specialmente perché esistono gli stessi segni fonetici che abbiano il significato diverso per differenti lettere. Per esempio il segno /dZi/ per la lettera g significa [gi] mentre per la lettera j significa [ji]. La notazione fonetica che si usa in dizionario ricevuto non è completamente originale, ma è un po' cambiata perché così la comprensibilità è più facile. A causa della differenza tra le due notazioni appaiono gli errori durante cambiamento dalla notazione ortografica alla quella fonetica. Alcuni errori sono presenti anche durante la divisone con le regole. L'analisi mostra le differenze tra la notazione ortografica e la notazione fonetica e percentuali degli errori sono: per le notazioni fonetiche e ortografiche: 77,29% di errore; per le notazioni ortografiche: 44,12% di errore; dopo l'analisi eseguita la quota di notazione corretta è 35,17%.

Le parole chiave: sillaba, sillabazione, notazione ortografica, notazione fonetica, errore, fonema

Sadržaj

1. Uvod

Zašto je slogovanje zaista bitno? Postoje dva stajališta koja daju odgovor na ovo pitanje. Prvo je lingvističko, a drugo je od strane računalne lingvistike. Računalna lingvistika je znanstvena disciplina koja se bavi razvojem računalnih programa koji omogućavaju obradu prirodnog jezika. Pod obradom prirodnog jezika možemo navesti ispravljanje tekstova, prijevod te interakciju između čovjeka i računala. Jedna od poddisciplina računalne lingvistike je prepoznavanje govora i sinteza govora koje proučavaju načine kako računalo zapravo može prepoznati, razumjeti i stvoriti prirodni jezik (Wikipedia, 2015).

Sintezu govora odnosno čitanje cijele riječi i analizu riječi odnosno rastavljanje riječi na slogove možemo analizirati od ranog razvoja djece na način da se promatra može li dijete iščitati cijelu riječ i da li je u stanju rastaviti riječ na slogove. Ako dijete pokazuje poteškoće sa sintezom i analizom trebalo bi se javiti logopedu (Šunić, 2008). Tu primjećujemo važnost slogovanja jer se ono koristi kod utvrđivanja disleksije, a i kod njezinog liječenja. Kod disleksije se javljaju poteškoće sa povezivanjem glasova i slogova u riječi, zamjenjuju se grafički ili fonetski slična slova, izostavljaju se slogovi i slova itd. (Šunić, 2008).

Možemo spomenuti i teoriju o evoluciji jezika koja ima utjecaj na evoluciju muzike i obrnuto. Jezik i muzička sposobnost baziraju se na nizu mogućnosti koje se javljaju i kod drugih primata, a ne samo kod ljudi. Kod istraživanju (Masataka, 2008) promatrali su se odrasli Japanski makaki-ji i ljudska novorođenčad te njihovo „dozivanje“ (engl. cooing). Novorođenčad ima sposobnost razlikovanja svih zvukova korištenih u svim jezicima, ali ne proizvodi zvukove. U ranoj fazi razvoja novorođenčad producira zvukove kao muziku i povezani su sa ritmičnim i melodičnim aspektima govora. Slična obilježja bila su primijećena i kod neljudskih primata kao što su Giboni i Japanski makaki-ji (Masataka, 2008). To rano pjevanje, dozivanje ili izgovaranje vokala kod djece može se povezati sa slogovima jer je to najmanja jedinica koja se može izgovoriti te se da zaključiti da je slogovanje jedna od elementarnih stvari od početka razvoja života.

Ovaj rad pokrivat će slogovanje talijanskog jezika koji je niske slogovne kompleksnosti. Smatra se da je talijanski jezik manje kompleksan i da ima jednostavniji slogovni sustav (Adsett, Marchand i Kešelj, 2009). Kad su djeca prvog i drugog razreda čitala riječi bez značenja ustanovila se razlika između kompleksnih i jednostavnih slogova u jezicima. Kod jezika sa jednostavnom slogovnom strukturom kao što su grčki, španjolski, francuski pa i talijanski

javljalo se puno manje pogrešaka kod čitanja nego kod jezika sa kompleksnim slogovnim sustavom kao što su engleski, njemački itd. (Adsett, Marchand i Kešelj, 2009).

U ovom istraživanju bit će objašnjena lingvistička teorija odnosno fonetska pravila i pravila za rastavljanje na slogove koja su temeljni dio rada. Rad će sadržavati i dio računalne teorije, istraživanja Adsett, Marchand i Kešelj. U četvrtom pooglavlju prikazat će se izlazne datoteke parsera koji je rađen u programu NetBeans IDE 8.0.2. a u zadnjem poglavlju bit će objašnjeni rezultati. Na kraju se nalazi i privitak, kod, pomoću kojeg su se parsirale i uspoređivale datoteke.

2. Lingvistička teorija

2.1 Osnovni pojmovi

Talijanska abedeca obuhvaća 21 slovo: a, b, c, d, e, f, g, h, i, l, m, n, o, p, q, r, s, t, u, v, z, a njima se pridodaje još 5 slova iz stranog porijekla: j, k, w, x, y. (Ziglio i Rizzo, 2004). Slova stranog porijekla upotrebljavaju se u tuđicama. Postoje još i združena slova izvan abecede, a to su: gli, gl (+i), gn, sc, sci, ch, gh, ci i gi (Jernej, 2012).

Grafički inventar talijanskog jezika kasnije će poslužiti za otkrivanje i analizu grešaka koje će se pojaviti u nastalim zapisima vraćanjem iz fonetskog u ortografski oblik.

Kako bi se lakše razumjela svrha i cilj ovog rada bitno je objasniti ključne riječi koje će se pojavljivati u ovom istraživanju.

Slog- element riječi formiran od jednog zvuka ili od više grupiranih zvukova (Zingarelli, 2008).

Slog- najmanja jedinica koju je moguće izgovoriti te je stoga i temeljna govorna jedinica (Škarić, 1991)

Slogovanje- rastavljanje riječi na slogove (Zingarelli, 2008).

Ortografski zapis- način točnog i normalnog zapisa pisanja riječi (Zingarelli, 2008).

Fonetika- nauka o zvukovima nekog jezika sa strane fizičkog gledišta (Zingarelli, 2008).

Fonetski zapis- zapis gdje svaki znak predstavlja jedan zvuk (Zingarelli, 2008).

Glas- najmanja govorna jedinica (samoglasnici i suglasnici) (Anić, 2007)., označava se uglatim zagradama [p] (Jozić, 2013).

Fonem- najmanja jezična jedinica koja ima razlikovnu vrijednost, a sama nema svoje značenje (Anić, 2007)., označava se kosim zagradama /p/ (Jozić, 2013).

2.2 Fonetska pravila

Tablica 1 Tablica suglasnika gdje jedno slovo predstavlja jedan glas (Šare, 2007)

Slovo	Simbol glasa u fonetskoj transkripciji
b	/b/
d	/d/
f	/f/
l	/l/
m	/m/
n	/n/
p	/p/
r	/r/
t	/t/
v	/v/

Tablica 1 prikazuje izgovor talijanskih suglasnika b, d, f, l, m, n, p, r, t, v koji je gotovo jednak izgovoru suglasnika u hrvatskom jeziku (Šare, 2007).

Tablica 2 Tablica suglasnika gdje jedno slovo predstavlja dva glasa (Šare, 2007)

Slovo	Simbol glasa u fonetskoj transkripciji
c	/k/
	/tʃ/
g	/g/
	/dʒ/
s	/s/

	/z/
z	/ts/
	/dz/

Tablica 2 prikazuje izgovor talijanskih suglasnika c, g, s i z koji kombinacijama pri pisanju daju različite rezultate pri izgovoru. Suglasnik c ispred e i i izgovara se č što je zapisano fonetskim znakom /tʃ/, a zapis ch izgovara se kao /k/ ispred e i i. Suglasnik g ispred e i i izgovara se dž što je zapisano fonetskim znakom /dʒ/, a zapis gh ispred e i i izgovara se /g/. Suglasnik s najčešće se izgovara kao hrvatsko s, između samoglasnika se izgovara kao z, ispred zvučnih suglasnika b, d, l, m, n, r, v izgovara se kao z. Suglasnik z se izgovara bezvučno kao hrvatsko c označeno fonetskim zapisom /ts/ ili kao zvučno dz označeno fonetskim zapisom /dz/ (Šare, 2007).

Tablica 3 Međunarodna fonetska abeceda (IPA) (Krämer, 2009)

CONSONANTS (PULMONIC) © 2005 IPA

	Bilabial	Labiodental	Dental	Alveolar	Postalveolar	Retroflex	Palatal	Velar	Uvular	Pharyngeal	Glottal
Plosive	p b			t d		ʈ ɖ	c ɟ	k g	q ɢ		ʔ
Nasal	m	ɱ		n		ɳ	ɲ	ŋ	ɴ		
Trill	ʙ			r					ʀ		
Tap or Flap		ⱱ		ɾ		ɽ					
Fricative	ɸ β	f v	θ ð	s z	ʃ ʒ	ʂ ʐ	ç ʝ	x ɣ	χ ʁ	ħ ʕ	h ɦ
Lateral fricative				ɬ ɮ							
Approximant		ʋ		ɹ		ɻ	j	ɰ			
Lateral approximant				l		ɭ	ʎ	ʟ			

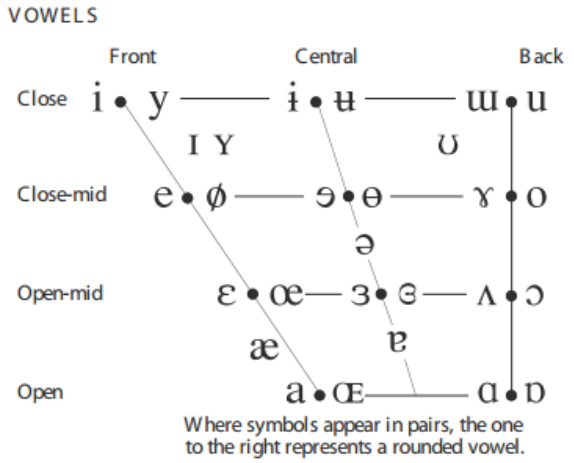
Where symbols appear in pairs, the one to the right represents a voiced consonant. Shaded areas denote articulations judged impossible.

Tablica 4 Međunarodna fonetska abedeca (IPA) (Krämer, 2009)

CONSONANTS (NON-PULMONIC)

Clicks		Voiced implosives		Ejectives	
ʘ	Bilabial	ɓ	Bilabial	ʼ	Examples:
ǀ	Dental	ɗ	Dental/alveolar	pʼ	Bilabial
ǃ	(Post)alveolar	ʄ	Palatal	tʼ	Dental/alveolar
ǂ	Palatoalveolar	ɠ	Velar	kʼ	Velar
ǁ	Alveolar lateral	ʛ	Uvular	sʼ	Alveolar fricative

Tablica 5 Međunarodna fonetska abedeca (IPA) (Krämer, 2009)



Tablice 3, 4 i 5 prikazuju međunarodnu fonetsku abecedu iz 2005. godine (IPA od engl. International Phonetic Alphabet), odnosno sustav znakova za fonološku i fonetsku transkripciju (Krämer, 2009).

Plućne suglasnike možemo vidjeti u tablici 3. Na mjestu gdje se simboli pojavljuju u parovima, desni predstavlja zvučni suglasnik, a lijevi bezvučni suglasnik. Tablica 4 sadržava pregradne suglasnike, a tablica 5 prikazuje samoglasnike, na mjestu gdje se simboli pojavljuju u parovima, onaj s desne strane predstavlja zaobljeni samoglasnik (Krämer, 2009).

2.3 Pravila rastavljanja riječi na slogove

Za rastavljanje riječi na slogove u talijanskom vrijede (Jernej, 2012) pravila:

1) Jednostavan suglasnik tvori slog sa sljedećim samoglasnikom. To vrijedi i kod složenica:
   - fa-ci-le,
   - do-ma-ni,
   - ma-la-ge-vo-le.
2) Dva različita suglasnika između dva samoglasnika pripadaju drugom slogu (ako prvi od njih nije l, m, n, r):
   - se-gre-to,
   - bo-sni-a-co,
   - re-pli-care.

Suglasničke skupine, u kojima suglasnici l, m, n, r dolaze na prvom mjestu, rastavljaju se tako da svaki od spomenuta četiri suglasnika čini slog s prethodnim samoglasnikom:

   - l'al-be-ro,
   - il com-pi-to.

3) Od tri različita suglasnika prvi pripada prethodnom slogu (ako to nije suglasnik s), a ostala dva slijedećem slogu:
   - sem-pre,
   - mi-san-tro-po.

To pravilo vrijedi i za sastavljene riječi:

   - tra-spor-ta-re,
   - e-sclu-de-re.

4) Dva se jednaka suglasnika rastavljaju:
   - il sof-fit-to,
   - op-pu-re.

U tu skupinu ide i suglasnik c ispred qu:

   - l'ac-qa,
   - ac-qui-sta-re.

5) Združena slova (digrami i trigrami) ne rastavljaju se:
   - o-gnu-no,

- la fa-mi-glia,
- il gior-na-le,

Združena slova u talijanskom jeziku nazivaju se digrami i trigrami.

Digrami: gn, gi, ci, ca, co, cu, ga, go, gu, ce, ge (Jernej, 2012).

Trigrami: gli, sci, che, chi, ghe, ghi, cia, cio, ciu, gia, gio, giu, sco (Jernej,2012).

6) Dvoglasi i troglasi ne rastavljaju se (ako to nisu „prividni dvoglasnici“):
   - la le-zio-ne,
   - l'e-sem-pio,
   - i-ta-lia-no.

Ali:

- il vi-o-li-no,
- a-dri-a-ti-co.

U talijanskom jeziku postoje diftonzi, odnosno dvoglasi, a većina njih za prvi član ima samoglasnike /i/ ili /u/. Javljaju se još i silazni dvoglasi kojima pripradaju sve ostale kombinacije vokala unutar istoga sloga (Šočanac, 2004).

Dvoglasi: io, ia, ie, oi, ai, ua, au, uo, eu, ue (Jernej, 2012).

Troglasi: iuo, iei, uoi (Jernej, 2012).

Slučajevi u kojima glasovi i, u, kad se nađu u riječi uz druge samoglasnike, zadržavaju vrijednost samoglasnika i ne čine diftong javlja se pojava koja se naziva hijat ili zijev, odnosno **prividni dvoglasnik**, tada se oba samoglasnika izgovaraju zasebno. Na primjer: tri-an-go-lo, vi-o-li-no, vi-a-dot-to, tri-on-fo. Malo koji rječnik u transkripciji ima zabilježeno polusamoglasno i, u, a pitanje polusamoglasnika i diftonga još nije dovoljno proučeno u talijanskom jeziku (Jernej, 2012).

3. Računalna znanost

Dobiveni format rječnika bio je napravljen za Italian Festival TTS project (Cosi, i dr., 2001), a sadrži pisani oblik riječi, značenje riječi, izgovor, naglasak i informacije o rastavljanju riječi na slogove.

Primjer preuzet iz (Adsett, Marchand i Kešelj, 2009):

(“sempre“ B (((s E1 m) ((pre) 0)))

- “sempre“ (talijanski 'uvijek') je riječ;
- B je značenje riječi, a označava prilog;
- “sE1mpre” je slijed fonema koji predstavljaju izgovor riječi → u ovom rječniku simbol ‘1’ označava naglašeni slog;
- “sE1m” je prvi slog;
- simbol 1 koji slijedi upućuje da ovaj slog dobiva primarni naglasak;
- “pre” je drugli slog (nije naglašen, što pokazuje broj 0).

U rječniku se nalazi 440084 zapisa, koji sadrže množinu riječi, sve glagolske oblike, superlative i komparative.

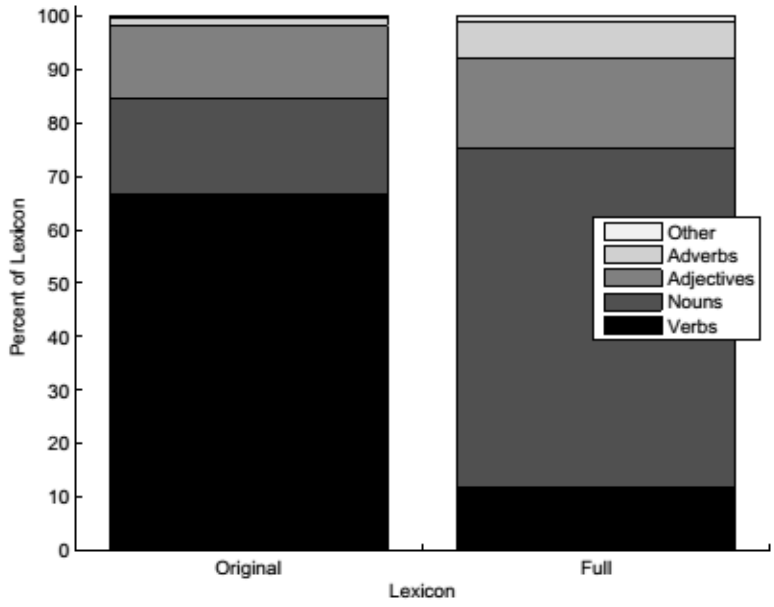


Figure 2: Distribution of parts of speech in the original and Full Italian lexicons.

Slika 1 Udio vrsta riječi u rječniku (Adsett, Marchand i Kešelj, 2009)

Na slici 1 prikazani su grafovi na kojima su vidljivi postoci vrsta riječi koje sadrži rječnik koji je korišten u ovom radu „Original“ i rječnik „Full“ koji je korišten u radu (Adsett, Marchand i Kešelj, 2009) gdje je smanjen broj riječi, odnosno maknute su množine riječi, komparativi, superlativi i ostavljen je samo infinitiv glagola.

U njihovom rječniku bila su testirana tri algoritma za automatsko slogovanje. Cioni algoritam za slogovanje pisanog talijanskog jezika, implementacija Hall-ovih pravila za rastavljanje talijanskih riječi na slogove i Bergamini-jev SYL-LABLE algoritam za rastavljanje riječi. Implementirana Cioni-jeva, Hall-ova i Bergamini-jeva pravila nalaze se u (Adsett, Marchand i Kešelj, 2009).

Cioni metoda sadrži 7 pravila:

1. CVCV → CV|CV;
2. VC_1C_2V → $VC_1|C_2V$, if $C_1=C_2$;
3. VC_1C_2V → $V|C_1C_2V$, if C_2=h;
4. VCCV → VC|CV;
5. $VC_1C_2C_3V$ → $VC_1|C_2C_3V$, if C_1 != s;
6. V_1V_2 → $V_1|V_2$, if $V_1 \in$ {a, e, o};
7. $V_1V_2V_3$ → $V_1|V_2V_3$ if $V_1 \in$ {a, e, o};

Hall-ova metoda sadrži 6 pravila:

1. C_1C_2 → $C_1|C_2$, if $C_1 = C_2$:
2. C_1C_2 → $C_1|C_2$, if C_1 = c and C_2 = q;
3. C_1C_2 → $C_1|C_2$, if $C_1 \in$ {m, n, l, r};
4. VCC → V|CC;
5. VCV → V|CV;
6. Never divide a sequence of vowels into multiple syllables.

Bergamini-jev program ima ova pravila:

1. V_1V_2 → V_1V_2, if $V_1 \in$ {a, e, o} and $V_2 \in$ {i, u};
2. V_1V_2 → V_1V_2, if $V_1 \in$ {i, u} and $V_2 \in$ {a, e, o};
3. $V_1V_2V_3$ → $V_1V_2V_3$, if $V_1V_2V_3$ ={iai, uai, iei, uei, iuo, uoi}:
4. V_1V_2 → $V_1|V_2$, in all other cases.
   Additional rules used include:

1. VCV → V|CV;
2. VC_1C_2 →$VC_1|C_2$, if $C_1 \epsilon$ {m, n, r, l};
3. C_1C_2 → $C_1|C_2$ if $C_1=C_2$;
4. C_1C_2 → $C_1|C_2$, if C_1 = c and C_2 = q;
5. VC_1C_2 →$V|C_1C_2$, in all other cases.

*C je simbol za suglasnik, a V je simbol za samoglasnik.

Rezultati (Adsett, Marchand i Kešelj, 2009) rastavljanja riječi na slogove prikazani su za svaku vrstu riječi posebno, a grafikon na slici 2 prikazuje postotke pogreške. Hall-ovim pravilom ustanovljeno je više 20% pogreške za imenice, manje od 20% za pridjeve te više od 10% Za glagole i priloge. Cioni-jevim pravilom ustanovljeno je oko 15% pogreške za imenice, oko 12% pogreške za pridjeve, manje od 10% pogreške za glagole i 10% pogreške za priloge. Bergamini-jev SYL-LABE algoritam ima oko 12% pogreške za imenice, oko 13% pogreške za pridjeve, 10% pogreške za glagole i oko 15% pogreške za priloge.

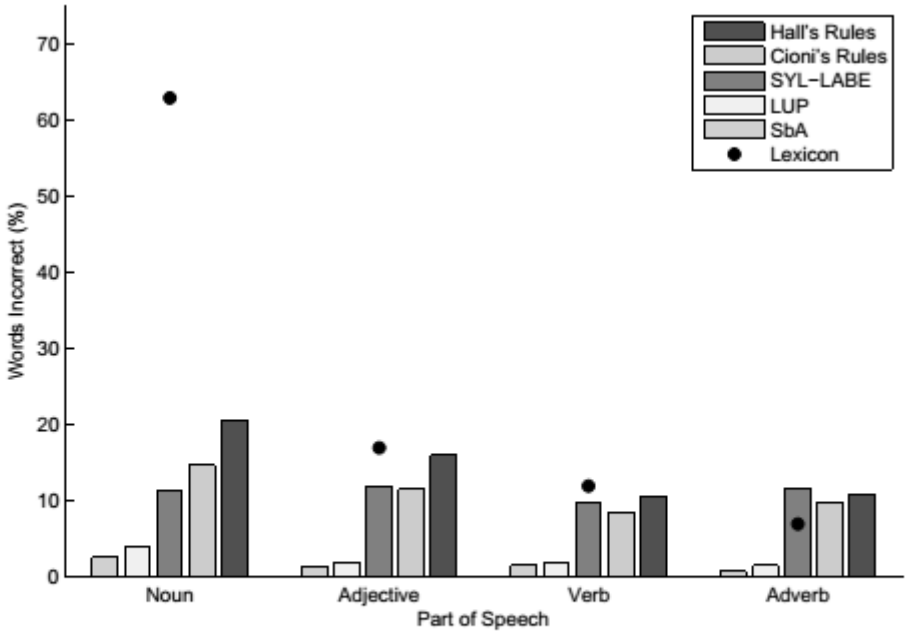


Figure 7: Word error percentages for each part of speech. The distribution of words as a function of parts of speech is given by the lexicon points. As with the previous figure, the Look-up Procedure results presented are those for the best performing version (version 10).

Slika 2 Postotak pogreške po vrstama riječi (Adsett, Marchand i Kešelj, 2009)

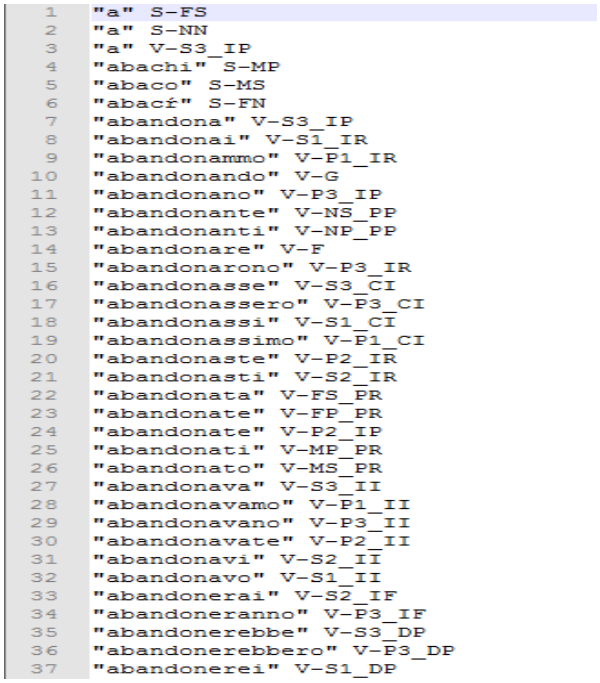
4. Parser

```
MNCL
("a" S-FS (((a1) 1)))
("a" S-NN (((a1) 1)))
("a" V-S3_IP (((a1) 1)))
("abachi" S-MP (((a1) 1) ((b a) 0) ((k i) 0)))
("abaco" S-MS (((a1) 1) ((b a) 0) ((k o) 0)))
("abací" S-FN (((a) 0) ((b a) 0) ((k a1) 1)))
("abandona" V-S3_IP (((a) 0) ((b a n) 0) ((d o1) 1) ((n a) 0)))
("abandonai" V-S1_IR (((a) 0) ((b a n) 0) ((d o) 0) ((n a1) 1) ((i) 0)))
("abandonammo" V-P1_IR (((a) 0) ((b a n) 0) ((d o) 0) ((n a1 m) 1) ((m o) 0)))
("abandonando" V-G (((a) 0) ((b a n) 0) ((d o) 0) ((n a1 n) 1) ((d o) 0)))
("abandonano" V-P3_IP (((a) 0) ((b a n) 0) ((d o1) 1) ((n a) 0) ((n o) 0)))
("abandonante" V-NS_PP (((a) 0) ((b a n) 0) ((d o) 0) ((n a1 n) 1) ((t e) 0)))
("abandonanti" V-NP_PP (((a) 0) ((b a n) 0) ((d o) 0) ((n a1 n) 1) ((t i) 0)))
("abandonare" V-F (((a) 0) ((b a n) 0) ((d o) 0) ((n a1) 1) ((r e) 0)))
("abandonarono" V-P3_IR (((a) 0) ((b a n) 0) ((d o) 0) ((n a1) 1) ((r o) 0) ((n o) 0)))
("abandonasse" V-S3_CI (((a) 0) ((b a n) 0) ((d o) 0) ((n a1 s) 1) ((s e) 0)))
("abandonassero" V-P3_CI (((a) 0) ((b a n) 0) ((d o) 0) ((n a1 s) 1) ((s e) 0) ((r o) 0)))
("abandonassi" V-S1_CI (((a) 0) ((b a n) 0) ((d o) 0) ((n a1 s) 1) ((s i) 0)))
("abandonassimo" V-P1_CI (((a) 0) ((b a n) 0) ((d o) 0) ((n a1 s) 1) ((s i) 0) ((m o) 0)))
("abandonaste" V-P2_IR (((a) 0) ((b a n) 0) ((d o) 0) ((n a1) 1) ((s t e) 0)))
("abandonasti" V-S2_IR (((a) 0) ((b a n) 0) ((d o) 0) ((n a1) 1) ((s t i) 0)))
("abandonata" V-FS_PR (((a) 0) ((b a n) 0) ((d o) 0) ((n a1) 1) ((t a) 0)))
("abandonate" V-FP_PR (((a) 0) ((b a n) 0) ((d o) 0) ((n a1) 1) ((t e) 0)))
("abandonate" V-P2_IP (((a) 0) ((b a n) 0) ((d o) 0) ((n a1) 1) ((t e) 0)))
("abandonati" V-MP_PR (((a) 0) ((b a n) 0) ((d o) 0) ((n a1) 1) ((t i) 0)))
("abandonato" V-MS_PR (((a) 0) ((b a n) 0) ((d o) 0) ((n a1) 1) ((t o) 0)))
```

Slika 3 Početna datoteka

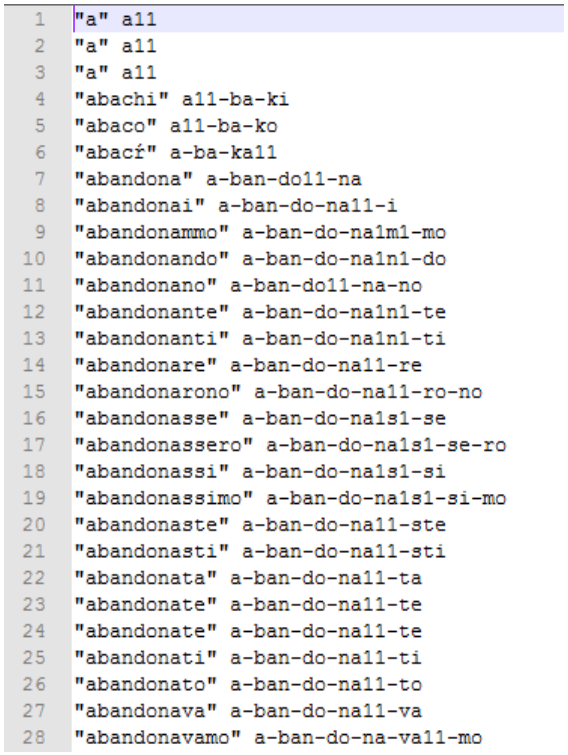
Početna datoteka podijeljena je u tri stupca. Prvi stupac sadržava riječ, drugi značenje riječi, a treći stupac riječ rastavljenu na slogove fonetskim znakovima. Prvi korak prema cilju bio je rastaviti datoteku u dvije datoteke, jednu u kojoj će se nalaziti riječ i njezino značenje i drugu u kojoj će biti riječ i ta ista riječ rastavljena na slogove. Rastavljanju datoteka prethodilo je parsiranje trećeg stupca na način da su se iz niza maknule zagrade i nule koje označavaju nenaglašeni slog, a umjesto zagrada dodavale su se povlake kako bi se lakše snalazilo između slogova. U parsiranom nizu ostale su jedinice koje označavanju naglašeni slog.

Iz niza (((a1) 1) ((b a) 0) ((k i) 0))) pomoću koda koji se nalazi u Privitku 1 dobio se niz a11-ba-ki.



```
"a" S-FS
"a" S-NN
"a" V-S3_IP
"abachi" S-MP
"abaco" S-MS
"abacś" S-FN
"abandona" V-S3_IP
"abandonai" V-S1_IR
"abandonammo" V-P1_IR
"abandonando" V-G
"abandonano" V-P3_IP
"abandonante" V-NS_PP
"abandonanti" V-NP_PP
"abandonare" V-F
"abandonarono" V-P3_IR
"abandonasse" V-S3_CI
"abandonassero" V-P3_CI
"abandonassi" V-S1_CI
"abandonassimo" V-P1_CI
"abandonaste" V-P2_IR
"abandonasti" V-S2_IR
"abandonata" V-FS_PR
"abandonate" V-FP_PR
"abandonate" V-P2_IP
"abandonati" V-MP_PR
"abandonato" V-MS_PR
"abandonava" V-S3_II
"abandonavamo" V-P1_II
"abandonavano" V-P3_II
"abandonavate" V-P2_II
"abandonavi" V-S2_II
"abandonavo" V-S1_II
"abandonerai" V-S2_IF
"abandoneranno" V-P3_IF
"abandonerebbe" V-S3_DP
"abandonerebbero" V-P3_DP
"abandonerei" V-S1_DP
```

Slika 4 Riječ i vrsta riječi



```
"a" a11
"a" a11
"a" a11
"abachi" a11-ba-ki
"abaco" a11-ba-ko
"abacś" a-ba-ka11
"abandona" a-ban-do11-na
"abandonai" a-ban-do-na11-i
"abandonammo" a-ban-do-na1m1-mo
"abandonando" a-ban-do-na1n1-do
"abandonano" a-ban-do11-na-no
"abandonante" a-ban-do-na1n1-te
"abandonanti" a-ban-do-na1n1-ti
"abandonare" a-ban-do-na11-re
"abandonarono" a-ban-do-na11-ro-no
"abandonasse" a-ban-do-na1s1-se
"abandonassero" a-ban-do-na1s1-se-ro
"abandonassi" a-ban-do-na1s1-si
"abandonassimo" a-ban-do-na1s1-si-mo
"abandonaste" a-ban-do-na11-ste
"abandonasti" a-ban-do-na11-sti
"abandonata" a-ban-do-na11-ta
"abandonate" a-ban-do-na11-te
"abandonate" a-ban-do-na11-te
"abandonati" a-ban-do-na11-ti
"abandonato" a-ban-do-na11-to
"abandonava" a-ban-do-na11-va
"abandonavamo" a-ban-do-na-va11-mo
```

Slika 5 Riječ i riječ rastavljena na slogove fonetskim znakovima

Nakon parsiranja bilo je potrebno implementirati pravila za rastavljanje riječi na slogove. U programu NetBeans IDE 8.0.2 pomoću jezika C++ implementirano je prvih pet pravila iz prijašnje cjeline. Pravilo šest nije implementirano iz razloga što postoje prividni dvoglasnici

koje je potrebno rastavljati, a njih se može prepoznati intuitivno što je problem kod računalne implementacije. Implementacija pravila vidljiva je u Privitku 1.

Nakon implementacije pravila bilo je potrebno rastaviti riječ iz prve kolone na slogove. Očito je da smo dobili zapis u ortografskom obliku. Cilj ovog rada je usporediti dva ortografska zapisa, pa je potrebno vratiti početni fonetski zapis u ortografski. To se radilo pomoću fonetskih pravila za svako slovo posebno. Fonetska pravila u prijašnjem poglavlju su pravila nulte razine, a u rječniku se nalaze i fonetski znakovi koji kazuju na kompromisnu transfonemizaciju[1] pa se kod vraćanja koristilo ovim promjenama (tablica 6):

Tablica 6 Promjene iz fonetskog zapisa u ortografski za sva slova

Fon.	Ort.	Fon.	Ort.
ki-	chi	tS	c
k-	c-	L-Li	-gli
ko	co	L-L	-gli
ka	ca	J-J	-gn
E	e	j	i
O	o	dz	z
nja	nia	ke	che
ts	z	w	u
kja	chia	dZ-	g-
dZi	gi	dZ	gi
S-S	-sci	S-Si	-sci

[1] Hrvatski i talijanski fonemski inventar imaju velike sličnosti te iz tog razloga nema potrebe za fonološkom adaptacijom posuđenica te replika zadržava fonološki oblik modela. U tom slučaju može se govoriti o nultoj transfonemizaciji, kod koje se fonemi jezika davatelja zamjenjuju odgovarajućim fonemima jezika primatelja čiji opis odgovara opisu fonema modela, odnosno samoglasnici se ne razlikuju po otvoru i mjestu artikulacije, a suglasnici po mjestu i načinu artikulacije, a kao primjer **kompromisne transfonemizacije** može se uzeti degeminacija. Na primjer kada se u hrvatskom jeziku talijanski fonem /b b/ mijenja sa hrvatskim /b/, /tts/ sa /c/, /ddʒ/ sa /đ/ i /dž/… (Sočanac, 2004).

k	c	dZe	ge

Neka slova imaju iste fonetske znakove, odnosno jedno slovo predstavlja dva različita glasa, zatim kod slova koja dolaze iz drugih jezika pojavljuju se nove promjene te zbog toga dolazi do ovih zamjena:

za slovo j prikaz je u tablici 7 i 8:

Tablica 7 Promjene iz fonetskog zapisa u ortografski za slovo j

Fon.	Ort.	Fon.	Ort.
dZa	ja	dZo	jo
dZu	ju	dZe	je
dZi	ji	i (na prvom mjestu)	j

tablica 7 prikazuje fonetske znakove koji su se mijenjali te njihove pripadajuće ortografske znakove kojima su se zamijenili kod zapisa za slovo j, a primjeri promijene mogu se vidjeti u tablici 8.

Tablica 8 Primjeri promjene iz fonetskog zapisa u ortografski za slovo j

FONETSKI	ORTOGRAFSKI	PRAVILIMA
dZak-k	jac-c	ja-c-ques
dZOn	jon	joh-n
dZu-ras-sik	ju-ras-sic	ju-ras-sic
dZE-jn	je-jn	ja-ne
i-vOn-ne	j-von-ne	jvon-ne
jadz-dzi-sta	jaz-zi-sta	jaz-zi-sta

Za slovo k prikaz je u tablici 9 i 10:

Tablica 9 Promjene iz fonetskog zapisa u ortografski za slovo k

Fon.	Ort.	Fon.	Ort.
S	sh	w	v
y	ni		

tablica 9 prikazuje primjere promjene kod slova k, a u tablici 10 vidljivi su i primjeri promjena.

Tablica 10 Promjene iz fonetskog zapisa u ortografski za slovo k

FONETSKI	ORTOGRAFSKI	PRAVILIMA
ka-laS-nni-kov	ka-lash-nni-kov	ka-la-shni-kov
kE-nja	ke-nia	ken-ya

Za slovo q prikaz je i tablici 11 i 12:

Tablica 11 Promjene iz fonetskog zapisa u ortografski za slovo q

Fon.	Ort.	Fon.	Ort.
q	k	kwi	qui
kwa	qua	kwo	quo
kwe	que		

tablica 11 prikazuje koje promjene iz fonetskog u ortografski zapis su se dešavale kod slova q, a u tablici 12 vidimo i primjere promjena.

Tablica 12 Primjeri promjene iz fonetskog zapisa u ortografski za slovo q

FONETSKI	ORTOGRAFSKI	PRAVILIMA
kwa-dra-no	qua-dra-no	qua-dra-no
kwel-lo	quel-lo	quel-lo
kwin-di	quin-di	quin-di
kwo-te-re-te	quo-te-re-te	quo-te-re-te

Za slovo w prikaz je u tablici 13 i 14:

Tablica 13 Promjene iz fonetskog zapisa u ortografski za slovo w

Fon.	Ort.	Fon.	Ort.
v	w	vu	w
S-S	sh		

tablica 13 prikazuje koje promjene su se koristile za slovo w, a tablica 14 sadržava neke primjere riječi za koje su se radile promjene.

Tablica 14 Primjeri promjene iz fonetskog zapisa u ortografski za slovo w

FONETSKI	ORTOGRAFSKI	PRAVILIMA
ve-ber	we-ber	we-ber
vu-vu-vu	w-w-w	w-w-w
wOS-Sin-ton	wo-shin-ton	wa-shin-gton

Za slovo y prikaz je u tablici 15 i 16:

Tablica 15 Promjene iz fonetskog zapisa u ortografski za slovo y

Fon.	Ort.	Fon.	Ort.
j	y	S	sh
i	y		

tablica 15 prikazuje promjene koje su se dešavale za zapise kod slova y, a tablica 16 sadržava primjere tih promjena.

Tablica 16 Primjeri promjene iz fonetskog zapisa u ortografski za slovo y

FONETSKI	ORTOGRAFSKI	PRAVILIMA
ja-le	ya-le	ya-le
jOr-kS-nga-ir	yor-ksh-nga-ir	yor-kshi-re
i-a-ma	y-a-ma	ya-ma-ha

Za slovo x prikaz je u tablici 17 i 18:

Tablica 17 Promjene iz fonetskog zapisa u ortografski za slovo x

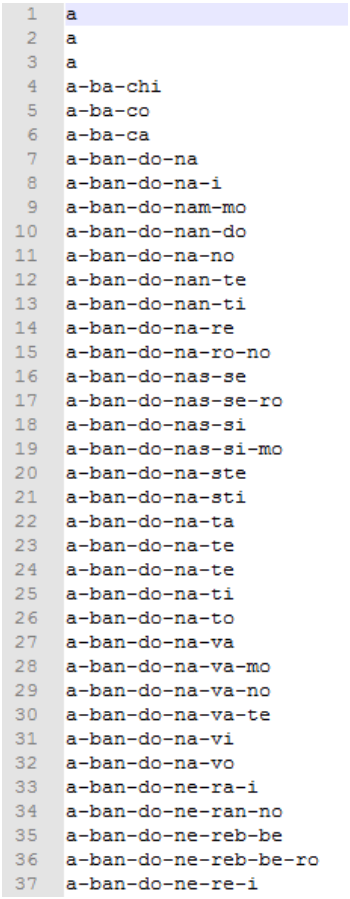
Fon.	Ort.
k-s	x

tablica 17 prikazuje koje promjene iz fonetskog u ortografski zapis su se dešavale kod slova x, a u tablici 18 vidimo i primjere promjena.

Tablica 18 Primjeri promjena iz fonetskog zapisa u ortografski za slovo x

FONETSKI	ORTOGRAFSKI	PRAVILIMA
k-se-nO-fo-bo	xe-no-fo-bo	xe-no-fo-bo

Nakon tih promjena dobivena je nova datoteka u kojoj se nalazila riječ iz početnog fonetskog zapisa promijenjena u ortografski zapis.



Slika 6 Ortografski zapis

Promjena iz fonetskog u ortografski zapis je bila posljednji korak prema usporedbi dva ortografska zapisa. Odnosno dva načina razdvajanja slogova. U slijedećem poglavlju prikazat će se rezultati dobiveni analizom.

5. Rezultati

Prva usporedba bila je napravljena između fonetskog i ortografskog zapisa, a dvije datoteke uspoređivale su se kodom koji se nalazi u Privitku 2. Uspoređeno je ukupno 440084 riječi, a pojavilo se 340160 pogrešaka. To znači da je tokom usporedbe pronađeno 77,29% pogreške i 22,71% točno rastavljenih riječi što se može vidjeti na slici 7.

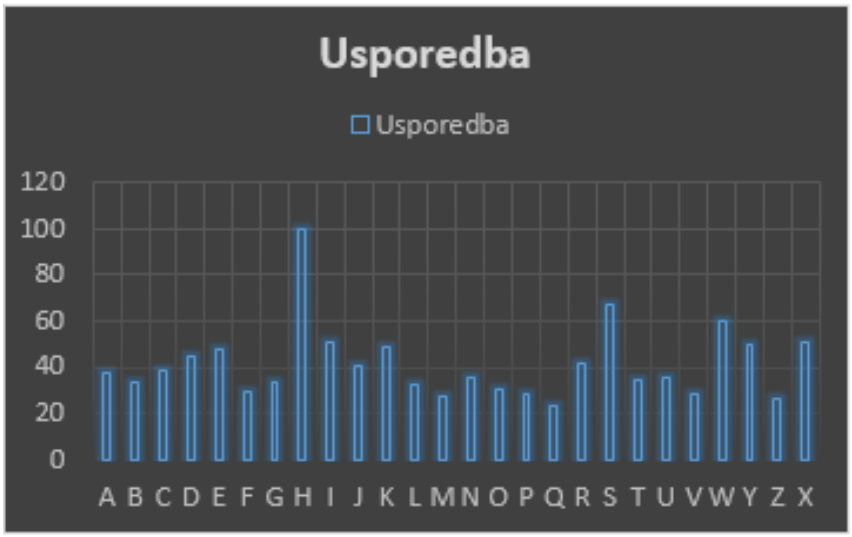
```
340162  Ukupno usporedenih rijeci : 440084
340163  Ukupno gresaka : 340160
340164  Postotak gresaka : 77.2943%
```

Slika 7 Rezultat pogreške između fonetskog i ortografskog zapisa

Kao što je rečeno prije, promjene iz fonetskog zapisa u ortografski zapis vršile su se za svako slovo posebno. Postoci pogreške dobiveni programom koji se nalazi u Privitku 3 prikazuju postotak pogrešaka koje su se desile kod svakog slova zasebno, a rezultati su vidljivi u tablici 19 i na grafikonu na slici 8.

Tablica 19 Postotak pogreške po slovima

A → 37,18%	B → 33,06%	C → 38,46%	D → 44,42%	E → 47,49%
F → 29,37%	G → 33,17%	H → 100%	I → 50,68%	J → 40,84%
K → 48,89%	L → 31,97%	M → 27,11%	N → 35,57%	O → 30,41%
P → 28,63%	Q → 23,59%	R → 41,65%	S → 67,25%	T → 34,02%
U → 35,15%	W → 60%	Y → 50%	Z → 25,94%	X → 50,98%
V → 28,50%				



Slika 8 Grafikon koji prikazuje postotak pogreške po slovima

Zadnji korak bila je usporedba ortografskog zapisa dobivenog implementacijom pravila i ortografskog zapisa vraćanog iz fonetskog oblika. Uspoređeno je 440084 riječi korištenjem koda iz Privitka 3, a analizom je utvrđena pogreška na 185357 riječi što iznosi 42,12% pogreške i 57,88% točno rastavljenih riječi (slika 9).

Nakon zadnjeg parsiranja ispravljeno je 35,17% zapisa.

```
370716   Ukupno usporedenih rijeci : 440084
370717   Ukupno gresaka : 185357
370718   Postotak gresaka : 42.1186%
```

Slika 9 Rezultat pogreške između dva ortografska zapisa

Pogreške:

1) Najveći postotak pogreške pojavio se kod slova H. Razlog tome je što se slovo H u talijanskom jeziku ne izgovara te se prilikom fonetskog zapisa ne zapisuje. Odnosno ne postoji fonetski znak koji bi označavao slovo H na početku riječi i zbog toga se slovo H na početku riječi ne može vraćati u ortografski zapis. Neki primjeri pogrešaka kod slova H su:

Ortografsko: an-di-kap-pa-no;
pravilima: han-di-cap-pa-no;

ortografsko: an-di-kap-pan-te;
pravilima: han-di-cap-pan-te;

ortografsko: an-di-kap-pan-ti;
pravilima: han-di-cap-pan-ti.

2) U setu dobivenih podataka pojavljuju se znakovi: ŕ, ň, é, ě, ů koji su fonetskim zapisom rastavljeni na a, o, e, i, u. Ti znakovi su vjerojatno kodirani prilikom prenošenja podataka, a označavaju talijanska slova sa naglaskom, odnosno slova à, ò, è, ì, ù. Tu dolazi do pogreške jer kod rastavljanja pravilima se uzima izvorna riječ te parser rastavlja nju na slogove, dok se kod vraćanja u ortografski zapis koristi već postojeći fonetski koji prepoznaje ta slova kao samoglasnike.

Ortografsko: a-ba-ca;
pravilima: a-bac-ŕ;

ortografsko: a-ban-do-no;
pravilima: a-ban-don-ň;

ortografsko: e;
pravilima: é;

ortografsko: i;
pravilima: ě;

ortografsko: u;
pravilima: ů.

3) Problem dvoglasa i troglasa, odnosno problem šestog pravila koje govori da se dvoglasi i troglasi ne rastavljaju osim u slučaju da se ne radi o „prividnim dvoglasima“. U kodu su implementirani dvoglasi i troglasi koje program prepoznaje te se zbog toga oni u ovom slučaju ne rastavljaju. Problem se javlja opet kod vraćanja iz fonetskog u ortografski zapis gdje su ti dvoglasi podijeljeni na slogove.

Ortografsko: ba-li-a;
pravilima: ba-lia;

ortografsko: fab-bri-che-re-i;
pravilima: fab-bri-che-rei;

ortografsko: na-bis-se-ra-i;
pravilima: na-bis-se-rai;

ortografsko: zu-ma-i;
pravilima: zu-mai.

4) Većina grešaka se dešava kod riječi koje su posuđene iz drugih jezika. Neke od tih grešaka možemo pridodati prije spomenutoj greški kod slova H koje se u talijanskom jeziku ne izgovara te se ni ne zapisuje fonetskim zapisom. Isto tako većina posuđenica se rastavlja po pravilima slogovanja jezika iz kojeg dolazi.

Ortografsko: nen-si;
pravilima: nan-cy;

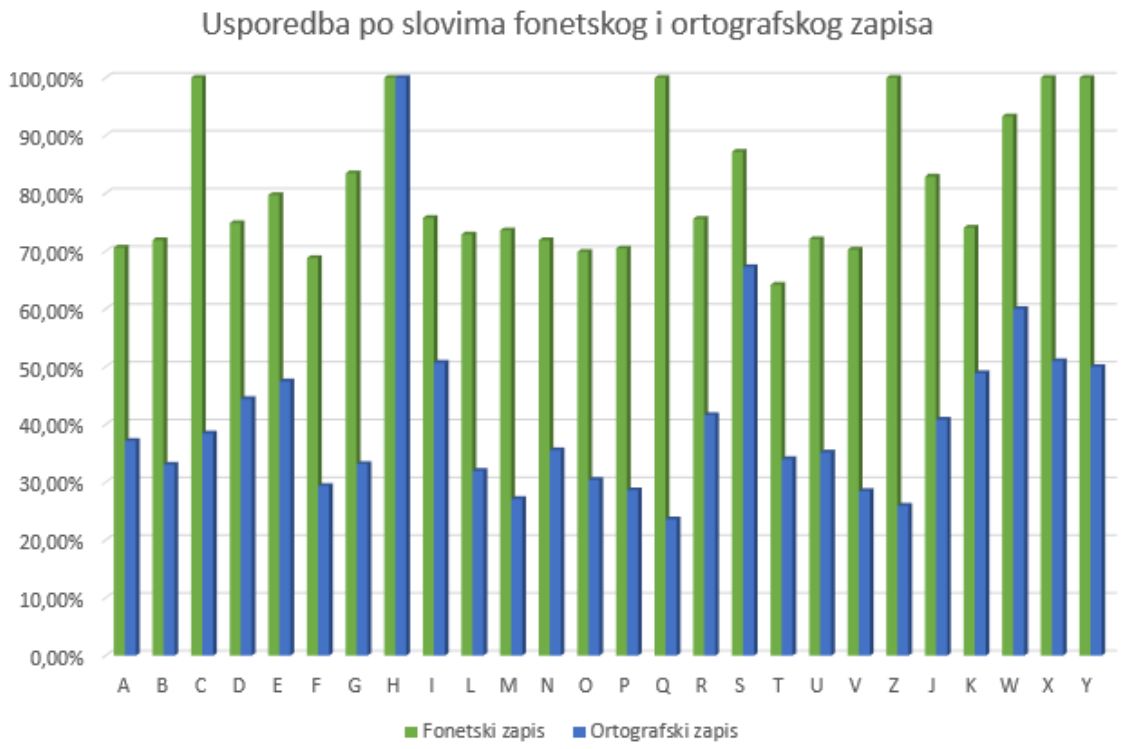
ortografsko: y-a-ma;
pravilima: ya-ma-ha;

ortografsko: wal-ter;
pravilima: wal-ther.

5) Promjena s u /z/ također je jedna od učestalijih pogrešaka. Kao što se vidi u poglavlju sa fonetskim pravilima slovo S u fonetskoj transkripciji se zapisuje sa /z/ i sa /s/. Zbog toga kod vraćanja u ortografski zapis nije točno sigurno koje /z/ se vraća u s, a koje ostaje z. Kad bi vraćali fonetsko /z/ u s, zamijenila bi se sva slova z pa čak i ona koja nije potrebno mijenjati.

Ortografsko: xa-u-za;
pravilima: xau-sa;

ortografsko: wa-le-za;
pravilima: wa-le-sa.



Slika 10 Usporedba po slovima između fonetskog zapisa i ortografskog zapisa

Grafikon na slici 10 prikazuje usporedbu između dva zapisa, ortografskog (prikazanog plavom bojom) i fonetskog (prikazanog zelenom bojom), za svako slovo zasebno. Promatrajući grafikon vidimo da se nakon parsiranja, zapis ispravio za većinu slova, a za neka i za više od 50%. Postoci ispravljenog zapisa vidljivi su u tablici 20.

Tablica 20 Postotak ispravljenog zapisa

A → 33,45%	B → 38,86%	C → 61,54%	D → 30,41%	E → 32,24%
F → 39,45%	G → 50,30%	H → 0%	I →2 5,06%	J → 42,07%
K → 25,18%	L → 40,93%	M → 46,50%	N → 36,33%	O → 39,46%
P → 41,83%	Q → 76,41%	R → 33,95%	S → 19,94%	T → 30,15%
U → 36,92%	W → 33,33%	Y → 50%	Z → 74,06%	X → 49,02%
V → 41,77%				

6. Zaključak

Jedna od važnih stvari jezika je pravilno rastavljanje riječi na slogove. Slogovanje se najčešće koristi u trenutku kada cijela riječ ne stane u jedan red kod pisanja rukom ili kod tiskanja novina te se mora rastaviti na slogove. Postoje generalna pravila za rastavljanje riječi na slogove, ali ponekad se znaju pojaviti sumnje kod neuobičajenih slučajeva (Serianni, 2005). Za rješavanje tih sumnji imamo pravila koja pomažu da slogovanje bude što točnije, a ona su navedena u prijašnjim poglavljima ovog rada. Slogovanje se koristiti i u logopediji kod liječenja nekih poremećaja kao što je disleksija, a koristi se i u računalnoj tehnici kod prepoznavanja govora te je prisutno od samog početka našeg života kada počinjemo sa učenjem jezika.

U ovom radu rastavljanje na slogove temeljilo se na zadanom rječniku talijanskog jezika. Parsiranjem, vraćanjem u ortografski zapis i implementiranjem pravila za slogovanje zapis se preoblikovao i mijenjao te je bio spreman za usporedbu dva ortografska zapisa. Prva usporedba iznosila je 77,29% pogreške, druga usporedba 42,12% pogreške. Vidimo da je nakon prvog parsiranja ispravljeno 35,17% zapisa.

Analizom su utvrđene greške koje se pojavljuju između dva ortografska zapisa za svako slovo posebno. Neke od učestalijih, koje se pojavljuju kod svih slova, navedene su u prethodnom poglavlju.

Uspoređivanjem grafikona sa slike 2 i slike 8 može se zaključiti da se i sa nekim rezultatima približilo rezultatu dobivenom kod istraživanja Adsett, Marchand i Kešelj, ali imajmo na umu da su na slici 2 prikazani rezultati pogreške podijeljeni prema vrstama riječi dok su rezultati na slici 8 prikazani za svako slovo posebno neovisno o vrstama riječi.

Smatram da bi u daljnjem radu bilo potrebno napraviti još nekoliko preinaka kako bi se još više smanjio postotak pogreške. Neke od mogućnosti su implementacija riječi u kojima se pojavljuju „prividni dvoglasi“ te posebna pravila po kojima bi se te riječi rastavljale. Zatim prepoznati koje slovo je točno kodirano znakovima: ŕ, ň, é, ě, ů te ih vratiti u odgovarajuće oblike. U početnom rječniku nalaze se i posuđenice koje imaju drugačiji izgovor i zapis. Trebalo bi proučiti pravila po kojima se te riječi rastavljaju na slogove. Uglavnom cilj bi bio pronaći rješenja za utvrđene pogreške.

7. Bibliografija

Adsett, Connie R., Yannick Marchand, i Vlado Kešelj. 2009. *Syllabification Rules Versus Data-driven Methods in a Language with Low Syllabic Complexity: The Case of Italian.* Nuova Scotia: Institute for Biodiagnostics (Atlantic) i Faculty of Computer science.

Anić, Vladimir. 2007. *Rječnik hrvatskoga jezika.* Zagreb: 2007.

Cosi, Piero, Fabio Tesser , Carlo Drioli, Graziano Tisato, i Roberto Gretter. 2005. *FESTIVAL in Italian.* 10. 3. Pokušaj pristupa 3. 3 2016. http://www2.pd.istc.cnr.it/FESTIVAL/home/default.htm.

Jernej, Josip. 2012. *Konverzacijska talijanska gramatika.* Zagreb: Školska knjiga.

Jozić, Željko. 2013. *Hrvatski pravopis.* Zagreb: Institut za hrvatski jezik i jezikoslovlje. Pokušaj pristupa 16. 7 2016. http://pravopis.hr/pravilo/tipovi-zagrada/66/.

Krämer, Martin. 2009. *The Phonology of Italian.* New York: Oxford University Press Inc.

Masataka, Nobuo. 2008. *The origins of language and the evolution of music: A comparative perspective.* Inuyama: Primate Research Institute.

Serianni, Luca. 2005. *Grammatica dell'italiano.* Torino: Redazione Garzanti.

Sočanac, Lelija. 2004. *Hrvatsko-talijanski jezični dodiri.* Zagreb: Nakladni zavod Globus.

Šare, Lidija. 2007. *Talijanska gramatika.* Split: Knjigotisak d.o.o.

Škarić, Ivo. 1991. *Fonetika hrvatskoga književnog jezika, u: Povijesni pregled, glasovi i oblici hrvatskoga književnog jezika (Nacrti za gramatiku).* Zagreb: Hrvatska akademija znanosti i umjetnosti i Nakladni zavod Globus.

Šunić, Nataša. 2008. *Kad slova ne slušaju...* 8. Pokušaj pristupa 15. 7 2016. http://www.vasezdravlje.com/printable/izdanje/clanak/1525/.

Wikipedia. 2015. *Računalna lingvistika.* 5. 8. Pokušaj pristupa 15. 7 2016. https://hr.wikipedia.org/wiki/Ra%C4%8Dunalna_lingvistika.

Ziglio, Luciana, i Giovanna Rizzo. 2004. *Espesso 1.* Firenze: Alma Edizioni.

Zingarelli, Nicola. 2008. *Vocabolario della lingua italiana.* Bologna: Zanichelli.

Privitak 1

U Privitku 1 nalazi se kod koji parsira i ispravlja početni set podataka, umeće povlake, briše zagrade, briše nule. U kodu su implementirana pravila za rastavljanje slogova.

```
#include <cstdlib>

#include <fstream>

#include <string>

#include <iostream>

#include <algorithm>

#include <ctype.h>

using namespace std;

char samoglasnici[]={'a', 'e', 'i', 'o', 'u'};

char sugl_lmnr[]={'l', 'm', 'n', 'r'};

string digrami [] = { "gn", "gi", "ci", "ca", "co", "cu", "ga", "go",
"gu", "ce", "ge"};

string trigrami [] = {"gli", "sci", "che", "chi", "ghe", "ghi", "cia",
"cio", "ciu", "gia", "gio", "giu", "sco"};

string dvoglasi [] = {"io", "ia", "ie", "oi", "ai", "ua", "au", "uo",
"eu", "ue"};

string troglasi [] = {"iuo", "iei", "uoi"};

/*funkcija koja dodaje minus na točno odredeno mjesto, prema pravilima
rastavljanja slogova poslije broja i zagrade, ali prije otvorene
zagrade */

string dodajMinuseke(string ulaz){

string izlaz = ulaz;

int brojac = 0;

while (brojac < izlaz.length()){

if(isdigit(izlaz[brojac])){//trenutni znak je broj

//ako je slijedeci zagrada, znaci da je kraj sloga

if(izlaz[brojac+1] == ')' && izlaz[brojac+3] == '('){

izlaz[brojac+2]='-';

brojac = brojac+2;

}

}
```

```
brojac++;
}
// cout << "minuseki su : " << izlaz << "\n";
return izlaz;
}
/* funkcija koja izbacuje razmake iz stringa*/
string izbaciRazmake(string ulaz){
int brojac = 0;
string temp;
while (brojac <= ulaz.length()){
if(ulaz[brojac] == ' '){
temp = ulaz.substr(0, brojac);
ulaz = temp + ulaz.substr(brojac+1);
}
brojac++;
}
// cout << ulaz << '\n';
return ulaz;
}
/* funkcija koja proalzi kroz niz znakova iz izbacuje otvorene i
zatvorene zagrade (()) */
string izbaciZagrade(string ulazniNiz){
string izlazniNiz;
string tempNiz  = "";
int duljina = ulazniNiz.length();
int pozicijaZagrade = 0;
// cout << ulazniNiz << '\n';
izlazniNiz = ulazniNiz;
while(duljina > 0){//za otvorene zagrade (
tempNiz  = "";
//cout << izlazniNiz << '\n';
pozicijaZagrade = izlazniNiz.find('(');
```

```
if(pozicijaZagrade > 1){//ako zagrada nije na prvom mjestu, moram sacuvati prvi dio stringa
tempNiz = izlazniNiz.substr(0, pozicijaZagrade);
}
izlazniNiz = tempNiz + izlazniNiz.substr(pozicijaZagrade+1);
duljina--;
}
//za zatvorene zagrade )
pozicijaZagrade = izlazniNiz.find(')');//uzimamo poziciju zagrade
do{
tempNiz  = "";
//zagrada nije na prvom mjestu, moram sacuvati prvi dio stringa
tempNiz = izlazniNiz.substr(0, pozicijaZagrade);
//ako je pozicija zagrade na kraju niza ili poslije kraja
if(pozicijaZagrade+1 >= izlazniNiz.length()){
izlazniNiz = tempNiz;
}
else{
izlazniNiz = tempNiz + izlazniNiz.substr(pozicijaZagrade+1);
}
//ponovo uzimamo poziciju zagrade ako je jos ima u nizu
pozicijaZagrade = izlazniNiz.find(')');
}while(pozicijaZagrade > 0);
//cout << izlazniNiz << '\n';
return izlazniNiz;
}
/*funkcija koja prolazi kroz ulazni niz znakova i izbacuje svako pojavljivanje nule (0) */
string izbaciNulice(string ulaz){
int duljinaNiza = 0;
string tempNiz;
while(duljinaNiza <= ulaz.length()){
```

```
if(ulaz[duljinaNiza] == '0'){//ako je trenutni znak 0

//izbacim ga van

tempNiz = ulaz.substr(0,duljinaNiza);

ulaz = tempNiz + ulaz.substr(duljinaNiza+1);

}

duljinaNiza++;

}

return ulaz;

}

/* funkcija vraca true ili false, ovisno da li je znak samoglasnik ili
ne */

bool samoglasnik(char znak){

int brojac = 0;

while(brojac <= 4){

if(znak == samoglasnici[brojac]){

return true;

}

brojac++;

}

return false;

}

/* funckija koja provjerava je li neka rijec digram */

bool digram(string rijec){

int brojac = 0;

while ( brojac < 10){

if(rijec == digrami[brojac]){

return true;

}

brojac ++;

}

return false;

}
```

```
/* funckija koja provjerava je li neka rijec trigram */
bool trigram(string rijec){
int brojac = 0;
while ( brojac <= 12){
if(rijec == trigrami[brojac]){
return true;
}
brojac ++;
}
return false;
}
/* funkcija vraca true ili false, ovisno da li je znak samoglasnik
l,m,n,r ili ne */
bool is_sugl_lmnr(char znak){
int brojac = 0;
while ( brojac <= 3){
if(znak == sugl_lmnr[brojac]){
return true;
}
brojac ++;
}
return false;
}
/* funckija koja provjerava je li neka rijec dvoglas */
bool dvoglas(string rijec){
int brojac = 0;
while ( brojac <= 9){
if(rijec == dvoglasi[brojac]){
return true;
}
brojac ++;
}
```

```
return false;
}

/* funckija koja provjerava je li neka rijec troglas*/
bool troglas(string rijec){
int brojac = 0;
while ( brojac <= 2){
if(rijec == troglasi[brojac]){
return true;
}
brojac ++;
}
return false;
}
//pravilo 1
string pravilo1(string rijec){
int brojac = 1;
string tempNiz;
//cout << "Rijec: " << rijec << "\n";
while(brojac < rijec.length()){
if(samoglasnik(rijec[brojac-1]) && !samoglasnik(rijec[brojac]) &&
samoglasnik(rijec[brojac+1])){
if(rijec[brojac-1] != '-'){//stavljamo minus, samo ako ga vec nema,
od nekog drugog pravila
tempNiz = rijec.substr(0, brojac);
rijec = tempNiz + '-' + rijec.substr(brojac);
}
brojac = brojac + 1;
}
brojac = brojac + 1;
//  cout<< "korak " << brojac << " rijec: " << rijec << "\n";
}
```

```
return rijec;

}

//Pravilo 2

string pravilo2(string rijec){

int brojac = 0;

string tempNiz;

//cout << "Rijec: " << rijec << "\n";

while(brojac < rijec.length()-3){

if(samoglasnik(rijec[brojac])                                          &&
samoglasnik(rijec[brojac+3])){//trenutni znak je samoglasnik, i za dva
dalje je isto samoglasnik

//sad treba provjeriti dal su unutarnja dva znaka suglasnici

if(!samoglasnik(rijec[brojac+1]) && !samoglasnik(rijec[brojac+2])  &&

 (rijec[brojac+1] != rijec[brojac+2])){

if(is_sugl_lmnr(rijec[brojac+1])){

tempNiz = rijec.substr(0, brojac+2);

rijec = tempNiz + '-' + rijec.substr(brojac+2);

brojac = brojac + 2;

}

 else{

tempNiz = rijec.substr(0, brojac+1);

rijec = tempNiz + '-' + rijec.substr(brojac+1);

brojac = brojac + 2;

}

}

}

brojac = brojac + 1;

//  cout<< "korak " << brojac << " rijec: " << rijec << "\n";

}

return rijec;

}

// Pravilo 3
```

```
string pravilo3(string rijec){

int brojac = 1;

string tempNiz;

//cout << "Rijec: " << rijec << "\n";

while(brojac <= rijec.length()-3){

//prvo treba vidjeti dal su tri suglasnika u nizu

if(!samoglasnik(rijec[brojac]    &&    !samoglasnik(rijec[brojac+1])
&& !samoglasnik(rijec[brojac+2])){

if(rijec[brojac]== 's'){

//Suglasnička skupina koja počinje slovom s, ne rastavlja se te pripada
iducem slogu.

if(rijec[brojac-1] != '-'){//stavljamo minus, samo ako ga vec nema,
od nekog drugog pravila

tempNiz = rijec.substr(0, brojac);

rijec = tempNiz + '-' + rijec.substr(brojac);

}

brojac = brojac+1;

}

else if(!samoglasnik(rijec[brojac])){//nije samoglasnik

//treba ispitati slijedeca dva znaka

if(!samoglasnik(rijec[brojac+1]) && !samoglasnik(rijec[brojac+2])){

if(rijec[brojac] != '-'){//stavljamo minus, samo ako ga vec nema, od
nekog drugog pravila

tempNiz = rijec.substr(0, brojac+1);

rijec = tempNiz + '-' + rijec.substr(brojac+1);

}

brojac = brojac + 2;

}

}

}

brojac = brojac + 1;

//  cout<< "korak " << brojac << " rijec: " << rijec << "\n";

}
```

```
return rijec;

}

//Pravilo 4

string pravilo4(string rijec){

int brojac = 0;

string tempNiz;

// cout<< "Pravilo4" << "\n";

//cout << rijec << "\n";

while(brojac < rijec.length()){

if(!samoglasnik(rijec[brojac])){//nije samoglasnik

if(rijec[brojac] == rijec[brojac+1]){//ako su dva suglasnika jedan do
drugoga

if(rijec[brojac] != '-'){

//stavljamo minus, samo ako ga vec nema, od nekog drugog pravila

tempNiz = rijec.substr(0, brojac+1);

rijec = tempNiz + '-' + rijec.substr(brojac+1);

}

brojac = brojac + 1;

}

//ako je c ispred qu

if(rijec[brojac] == 'c' && rijec[brojac+1] == 'q' && rijec[brojac+2]
== 'u'){

if(rijec[brojac] != '-'){

//stavljamo minus, samo ako ga vec nema, od nekog drugog pravila

tempNiz = rijec.substr(0, brojac+1);

rijec = tempNiz + '-' + rijec.substr(brojac+1);

}

brojac = brojac +1;

}

}

brojac = brojac + 1;

}
```

```
return rijec;

}

//Pravilo 5

string pravilo5(string rijec){

int brojac = 2; //nema smisla provjeravati digram i trigram na prvom
slovu

string tempNiz;

while(brojac <= rijec.length()){

if(digram(rijec.substr(brojac,2))){//znaci da je tu digram

if(rijec[brojac-1] != '-'){

//stavljamo minus, samo ako ga vec nema, od nekog drugog pravila

tempNiz = rijec.substr(0,brojac);

rijec = tempNiz + '-' + rijec.substr(brojac);

}

brojac = brojac + 1;

}

else if(trigram(rijec.substr(brojac,3))){//znaci da je tu digram

if(rijec[brojac-1] != '-'){//ako jos ne postoji minus, stavi ga

tempNiz = rijec.substr(0,brojac);

rijec = tempNiz + '-' + rijec.substr(brojac);

}

brojac = brojac + 2;

}

brojac ++;

}

return rijec;

}

//Pravilo 6

string pravilo6(string rijec){

}

/* ovo je funkcija koja će pozivati svako pojedino pravilo i na taj
rnacin rastavljati rijeci*/
```

```
string pravila(string rijec){
string rezultat;
rezultat = pravilo3(rijec);
//cout << "pravilo 3: " << rezultat << "\n";
rezultat = pravilo2(rezultat);
//cout << "pravilo 2: " << rezultat << "\n";
rezultat = pravilo4(rezultat);
//cout << "pravilo 4: " << rezultat << "\n";
rezultat = pravilo5(rezultat);
//cout << "pravilo 5: " << rezultat << "\n";
rezultat = pravilo1(rezultat);
//cout << "pravilo 1: " << rezultat << "\n";
return rezultat;
}
int main() {
string ulaznaLinija;
string prvakolona;
string drugakolona;
string trecakolona;
string linija;
string parsiraniNiz;
string parsiraniNiz1;
string parsiraniNiz2;
string parsiraniNiz3;
int brojac = 1;
//ulazna datoteka
//  ifstream ulaznaDatoteka("lexprvih99.out");
ifstream ulaznaDatoteka("lex.out");
if (ulaznaDatoteka.is_open()) {
//izlazan datoteka 1
//ofstream znacenjeDatoteka("prvih99znacenje.out");
```

```
ofstream znacenjeDatoteka("znacenje.out");
//izlazna datoteka 2
//ofstream slogoviDatoteka("prvih99slogovi.out");
ofstream slogoviDatoteka("slogovi.out");
// ofstream praviloDatoteka("praviloSlogovi99.out");
ofstream praviloDatoteka("praviloSlogovi.out");
ofstream trecaKolona("trecaKolona.out");
//cout << "pravila: " << pravila("abandonai") << "\n";
/**/
while (getline(ulaznaDatoteka, ulaznaLinija)) {
prvakolona = "";
drugakolona = "";
trecakolona = "";
linija = "";
if (brojac != 1) { //prva linija je neki slog koji preskacemo
//micemo prvu i zadnju zagradu
linija = ulaznaLinija.substr(1, ulaznaLinija.length()-3);
//2 znaka su oznaka za novi red i jedna zagrada
//sve do prvog razmaka ide u prvu kolonu
prvakolona = linija.substr(0, linija.find(' '));
//micem (cut-am) prvu kolonu iz ulaznog stringa
linija = linija.substr(prvakolona.length() + 1);
//druga kolona je sve do slijedeceg razmaka
drugakolona = linija.substr(0, linija.find(' '));
//treca kolona je sve preostalo
trecakolona = linija.substr(drugakolona.length() + 1);
// cout << prvakolona << '\n';
//cout << drugakolona << '\n';
//cout << trecakolona << '\n';
//dodajemo minuse u trecu kolonu
parsiraniNiz = dodajMinuseke(trecakolona) ;
```

```
//   cout << "dodani minuseki: " << parsiraniNiz << '\n';
//izbacujemo razmake iz trece kolone
parsiraniNiz1 = izbaciRazmake(parsiraniNiz) ;
//   cout << "izbaceni razmakici " << parsiraniNiz1 << "\n";
//izbacujemo zagrade van
parsiraniNiz2 = izbaciZagrade(parsiraniNiz1);
//  cout << "izbacene zagrade " << parsiraniNiz2 << "\n";
//jos trebamo izbaciti nule
parsiraniNiz3 = izbaciNulice(parsiraniNiz2);
//cout << "izbacene nulice " << parsiraniNiz2 << "\n";
if (znacenjeDatoteka.is_open()) {
znacenjeDatoteka << prvakolona;
znacenjeDatoteka << ' ';
znacenjeDatoteka << drugakolona;
znacenjeDatoteka << '\n';
}
else{
cout << "Nije moguce otvoriti datoteku";
}
if (slogoviDatoteka.is_open()) {
slogoviDatoteka << prvakolona;
slogoviDatoteka << ' ';
slogoviDatoteka << parsiraniNiz3;
slogoviDatoteka << '\n';
}
else{
cout << "Nije moguce otvoriti datoteku";
}
if (praviloDatoteka.is_open()) {
praviloDatoteka << prvakolona;
praviloDatoteka << ' ';
```

```
praviloDatoteka << pravila(prvakolona);
praviloDatoteka << '\n';
}
else{
cout << "Nije moguce otvoriti datoteku";
}
if (trecaKolona.is_open()) {
trecaKolona << parsiraniNiz3;
trecaKolona << '\n';
}
else{
cout << "Nije moguce otvoriti datoteku";
}
}
brojac++;
}
ulaznaDatoteka.close();
znacenjeDatoteka.close();
}
else {
cout << "Datoteku nije moguce otvoriti";
}
return 0;
}
```

Privitak 2

Ovaj kod služi za uspoređivanje dvije datoteke. Uspoređuje se datoteka u kojoj se nalazi fonetski zapis te datoteka u kojoj se nalazi ortografski zapis.

```
#include <cstdlib>

#include <string>

#include <iostream>

#include <fstream>

using namespace std;

int main() {

string ulaznaLinija1;//citamo redak iz prve datoteke

string ulaznaLinija2;//citamo redak iz druge datoteke

int razmak;

string drugakolona;

double ukupnoRedaka = 0;

double brojGresaka = 0;

//ulazna datoteka

ifstream ulaznaPravilo("praviloSlogovi.out");//prva datoteka

ifstream ulaznaTrecaKolona("trecaKolona.out");//druga datoteka

if (ulaznaPravilo.is_open() && ulaznaTrecaKolona.is_open()) {

//izlazan datoteka

ofstream usporedbaDatoteka("rezultatUsporedbe.out");

 /**/

while (getline(ulaznaPravilo, ulaznaLinija1)) {//dohvacamo redak iz
prve datoteke

//dohvacamo redak iz druge datoteke

getline(ulaznaTrecaKolona, ulaznaLinija2);

razmak = ulaznaLinija1.find(' ');

drugakolona = ulaznaLinija1.substr(razmak+1);

drugakolona = drugakolona.substr(1, drugakolona.length()-2);

//cout << "druga kolona " << drugakolona << "\n";

if(ulaznaLinija2.compare(drugakolona) != 0){
```

```
brojGresaka ++;

if(brojGresaka == 1){

cout << "Razlicite rijeci: " << "\n";

usporedbaDatoteka << "Razlicite rijeci: " << "\n";

}

cout << "Ortografsko: " << ulaznaLinija2 << " Zadana: " << drugakolona
<<  "\n";

usporedbaDatoteka << "Ortografsko: " << ulaznaLinija2 << " Zadana: "
<< drugakolona <<  "\n";

}

ukupnoRedaka ++;

}

ulaznaPravilo.close();

ulaznaTrecaKolona.close();

cout << "Ukupno usporedenih rijeci : " << ukupnoRedaka <<"\n";

cout << "Ukupno gresaka : " << brojGresaka <<"\n";

cout << "Postotak gresaka : " << (brojGresaka/ukupnoRedaka)*100 <<"%
\n";

usporedbaDatoteka << "Ukupno usporedenih rijeci : " << ukupnoRedaka
<<"\n";

usporedbaDatoteka << "Ukupno gresaka : " << brojGresaka <<"\n";

usporedbaDatoteka     <<     "Postotak     gresaka     :     "     <<
(brojGresaka/ukupnoRedaka)*100 <<"% \n";

}

else {

cout << "Datoteke nije moguce otvoriti";

}

return 0;

}
```

Privitak 3

U privitku 3 nalazi se kod koji također uspoređuje dvije datoteke. Kod uspoređuje sve datoteke po slovima te na kraju datoteku u kojoj se nalazi ortografski zapis vraćan iz fonetskog i ortografski zapis dobiven rastavljanjem riječi po pravilima.

```
#include <cstdlib>

#include <string>

#include <iostream>

#include <fstream>

using namespace std;

int main() {

string ulaznaLinija1;//citamo redak iz prve datoteke

string ulaznaLinija2;//citamo redak iz druge datoteke

int razmak;

//string drugakolona;

double ukupnoRedaka = 0;

double brojGresaka = 0;

//ulazna datoteka

ifstream SvePravilima("SvePravilima.txt");//prva datoteka

ifstream SveOrtografski("SveOrtografski.txt");//druga datoteka

if (SvePravilima.is_open() && SveOrtografski.is_open()) {

//izlazna datoteka

ofstream usporedbaDatoteka("rezultatUsporedbeOrtografskih.out");

/**/

while (getline(SvePravilima, ulaznaLinija1)) {//dohvacamo redak iz
prve datoteke

//dohvacamo redak iz druge datoteke

getline(SveOrtografski, ulaznaLinija2);

if(ulaznaLinija2.compare(ulaznaLinija1) != 0){

brojGresaka ++;

if(brojGresaka == 1){

cout << "Razlicite rijeci: " << "\n";

usporedbaDatoteka << "Razlicite rijeci: " << "\n";
```

```
}

cout << "Ortografsko: " << ulaznaLinija2 << " Pravilima: " <<
ulaznaLinija1 <<  "\n";

usporedbaDatoteka << "Ortografsko: " << ulaznaLinija2 << " Pravilima:
" << ulaznaLinija1 <<  "\n";

 }

ukupnoRedaka ++;

}

SvePravilima.close();

SveOrtografski.close();

cout << "Ukupno usporedenih rijeci : " << ukupnoRedaka <<"\n";

cout << "Ukupno gresaka : " << brojGresaka <<"\n";

cout << "Postotak gresaka : " << (brojGresaka/ukupnoRedaka)*100 <<"%
\n";

usporedbaDatoteka << "Ukupno usporedenih rijeci : " << ukupnoRedaka
<<"\n";

usporedbaDatoteka << "Ukupno gresaka : " << brojGresaka <<"\n";

usporedbaDatoteka     <<     "Postotak     gresaka     :     "     <<
(brojGresaka/ukupnoRedaka)*100 <<"% \n";

}

else {

cout << "Datoteke nije moguce otvoriti";

}

    return 0;

}
```