VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV POČÍTAČOVÉ GRAFIKY A MULTIMÉDIÍ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF COMPUTER GRAPHICS AND MUTIMEDIA

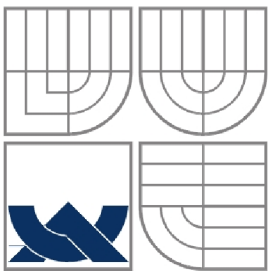
# MORFOLGICKÝ ANALYZÁTOR POMOCOU KONEČNÝCH AUTOMATOV

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

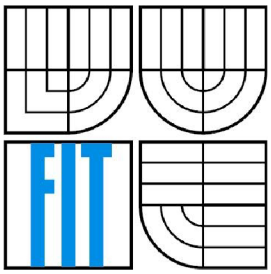
AUTOR PRÁCE
AUTHOR

JAKUB BEZÁK

BRNO 2009



VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV POČÍTAČOVÝCH SYSTÉMŮ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF COMPUTER GRAPHICS AND MULTIMEDIA

# MORFOLGICKÝ ANALYZÁTOR POMOCÍ KONEČNÝCH AUTOMATŮ

MORPHOLOGICAL ANALYZER IMPLEMENTED AS FSA

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE
AUTHOR

JAKUB BEZÁK

VEDOUCÍ PRÁCE
SUPERVISOR

doc. RNDr. PAVEL SMRŽ, Ph.D.

BRNO 2009

# Abstrakt

Cílem tohoto textu je čtenáři co nejvíc přiblížit problematiku morfologie a morfologické analýzy slovenského jazyka s využitím morfologického analyzátoru a nastínit odlišnost od různých světových jazyků. Text také objasňuje význam hledání základního tvaru slov při analýze slovenských textů. V další části se věnuje optimálnímu uložení rozsáhlých slovníků pomocí konečných automatů s využitím jejich minimalizace. Nakonec jsou srovnávány různé formy implementace.



# Abstract

The aim of this paper is to introduce to its reader the field of morphemics and morphemic analysis of slovak language using morphemic analyser and to signify the difference from other world languages. In this text it is also explained the importance of finding base form of words while analysing slovak texts. Other part analyses the optimal storage of large dictionaries using finite state automata minimization. The end contains the comparison between various forms of implementation.



# Klíčová slova

morfologie, morfologická analýza, konečné automaty, minimalizace konečných automatů, prirozené spracování jazyka

# Keywords

morphemics, morphemic analysis, finite state automata, minimization of finite state automata, natural language processing


# Citace

Jakub Bezák: Morfologický analyzátor pomocou konečných automatov, bakalářská práce, Brno, FIT VUT v Brně, 2009

# Morfologický analyzátor pomocou konečných automatov

## Prehlásenie

Prehlasujem, že som túto bakalársku prácu vypracoval samostatne pod vedením doc. Pavla Smrža. Dalšie informácie mi poskytli kolegovia z projektu NLP. Uviedol som všetky literárne pramene a publikáce, z ktorých som čerpal.

.....
Jakub Bezák
20. mája 2009

## Poďakovanie

Týmto by som chcel poďakovať môjmu vedúcemu doc. Pavlovi Smržovi za jeho rady a podnety. Ďalej by som chcel poďakovať spolupracovníkom z NLP za poskytnuté dáta, obzvlášť Stanislavovi Černému za rady ohľadom transformácií slovníkov.

# Obsah

# 1 Úvod

Jazyk je dôležitým prostriedkom komunikácie, či už v písanej alebo ústnej podobe. Jeho vznik sa datuje od počiatku ľudstva, a aj keď v súčasnosti má už každý jazyk pevné základy, stále je živý a vyvíja sa. Je žiadúce tieto zmeny zaznamenávať, udržiavať a nové, ale aj pôvodné slová analyzovať na základe morfológie, lexiky, syntaxe a štylistiky.

Táto bakalárska práca sa zaoberá jazykom z pohľadu morfologickej analýzy. Jednotlivé slová jazyka spracúva morfologický analyzátor, ktorý by mal vedieť z daného slova určiť gramatické kategórie a základný tvar daného slova.

Práca nadväzuje na predchádzajúci výskum v skupine prirodzeného spracovanie jazyka NLP – *Natural language processing*, z ktorého vzišli morfologické analyzátory českého jazyka *ajka* [8][9] a *ma* [4]. Keďže navrhovaný morfologický analyzátor bude slúžiť na analýzu slovenčiny, bude ďalej rozoberaný slovenský jazyk, ak to nebude uvedené inak.

V kapitole 2 Morfológia sú uvedené a ozrejmené základné pojmy tejto lingvistickej vednej disciplíny a postupy pri ohýbaní a tvorení nových slov, s ktorými by sa mal čitateľ tejto práce oboznámiť, pre lepšie pochopenie problematík rozoberaných v ďalších kapitolách.

Na otázku, či je analýza slovenského jazyka komplikovaná, dáva odpoveď kapitola 3 Morfologický analyzátor, kde je slovenčina čiastočne porovnaná s germánskymi jazykmi angličtinou a nemčinou a románskym jazykom španielčinou. Ďalej je spomenutý proces, ktorý dokáže z hociktorého tvaru slova dostať tvar základný, a tým uľahčiť celý proces analýzy.

Kapitola 4 Uloženie dát skúma možnosti uloženia slovníkov v dátových štruktúrach podobným konečným automatom a tiež možnosti využitia minimalizácie konečných automatov na minimalizovanie veľkosti slovníkov.

Na túto kapitolu nadväzuje kapitole 5 Príprava dát, ktorá popisuje postupy pri tvorbe systému vzorov slovenských dát a ich transformáciu na formát vhodný pre morfologický analyzátor.

V kapitole 6 Implementácia sa nachádza popis funkcií morfologického analyzátora, porovnanie s inými aplikáciami na základe vykonaných testov a vyhodnotenie testov.

# 2 Morfológia

Morfológia, čiže tvaroslovie je jazykovedná náuka o gramatických tvaroch slov, ako aj o slovách, ktoré majú funkciu tvarov. Morfológia je teda náuka o tvarovej rovine v systéme jazyka. Táto rovina je vybudovaná na slovných druhoch, na morfologických kategóriách a na rozličných tvarotvorných postupoch [1].

Inými slovami, morfológia sa zaoberá slovnými druhmi, ohýbaním slovných druhov rôznymi spôsobmi: skloňovaním (podstatné mená, prídavné mená, zámena, číslovky: pes – psa), časovaním (slovesá: písať – píše) alebo stupňovaním(prídavné mená, príslovky: trpezlivo – trpezlivejšie) a tvorením slov napríklad odvodzovaním (učiť – učiteľka ), skladaním (bez vedomia – bezvedomie), prechod medzi slovnými druhmi (domáci (prídavné meno) – domáci (podstatné meno)) a tvorením citovo zafarbených slov (ruka – ručička (citovo zjemňujúce), ruka – ručisko (citovo zhoršujúce)).

## 2.1 Ohýbanie slov

### 2.1.1 Pojmy

**morféma**: najmenšia časť slova, ktorá má význam alebo funkciu
**lexikálna morféma**: časť slova, ktorá udáva jeho význam
**gramatická morféma**: vyjadruje vzťah medzi slovami
**modifikačná morféma**: časť medzi lexikálnou a gramatickou morfémou
**tvarotvorný základ (báza):** súhrn lexikálnej a modifikačnej morfémy
**tvorotvorná prípona (formant):** pridáva sa k slovu pri ohýbaní, mení tvar slova, význam ostáva

### 2.1.2 Gramatické tvary

Gramatický tvar vyjadruje gramatický význam slova vonkajšími jazykovými prostriedkami. Každý gramatický tvar sa skladá z morfém. Rozlišujeme jednomorfné tvary (nedajú sa členiť na menšie jednotky), ktoré nie sú veľmi časté a viacmorfné tvary. V každom viacmorfnom tvare sa skladá aspoň z lexikálne morfémy, ktorá určuje jeho význam, a gramatickej morfémy, ktorá určuje jeho formu [1].

V tvaroch so zložitejšou štruktúrou rozlišujeme dve časti: tvarotvorný základ a tvarotvornú príponu. Tvarotvorný základ najčastejšie ostáva nezmenený vo všetkých tvaroch slov. Niektoré zmeny sa prejavujú len v zvukovej podobe, čo platí napríklad pre spoluhlásky *d,t,n,l*, po ktorých nasleduje samohláska *e* alebo *i*, sa mení zvuková podoba na *ď, ť, ň, ľ*. Iné zmeny v tvarotvornom

základe, ktoré nezávisia na tvarotvornej prípone sa nazývajú alternácie. Napríklad k → c, ch → s, c → č [1].

**Príklad 2.1:** Zmeny v tvaroslovnom základe

**Zvuková podoba:**

tunel – v tuneli, hrad – na hrade, päta – v päte, pán – pane

**Alternácie:**

Čech – Česi, vták – vtáci

Výnimočnými sú veľké variácie v báze, napríklad v slove hnať, pri ktorom sa v rôznych tvaroch používa tvarotvorný základ *hna-* a *žen-*, prípadne v slove jať – (za)*jmem*. Tieto tvary už majú málo spoločné a blížia sa k tzv. supletívným (dopĺňajúcim) tvarom, pri ktorých sa jednotlivé tvary tvoria od rôznych tvarotvorných základov. V slovenčine je takýchto tvarov pomerne málo. Typickým príkladom je podstatné meno *človek*, ktorého plurál sa tvorí od slovotvorného základu *ľud-* (ľudia, ľudí,...). Ďalšími zástupcami supletívnosti sú nepravidelne stupňované prídavné mená (dobrý, lepší, najlepší) a príslovky (pekne, krajšie, najkrajšie) alebo osobné zámená (ja – mňa, my – nás, oni – ich, ...).

Druhou časťou gramatického tvaru je tvarotvorná prípona, ktorá je nerozložiteľná. Viditeľné je opakovanie sa tvarotvorných prípon pre istý jav v slovných druhoch alebo pre istú skupinu slov slovného druhu. Napríklad prípona *-m* vyjadruje gramatický tvar 1. osoby jednotného čísla prítomného času slovies a prípona *-y* môže vyjadrovať nominatív množného čísla niektorých podstatných mien (skloňovaných podľa vzoru žena, dub alebo idea).

Tvorenie všetkých gramatických tvarov daného slova sa nazýva ohýbanie slov. Ohýbanie môžeme rozdeliť do troch kategórií: *skloňovanie*, čiže tvorenie pádových tvarov podstatných a prídavných mien, zámen a čísloviek; *časovanie*, čiže tvorenie slovesných tvarov slovies, ktoré vyjadrujú čas a spôsob; *stupňovanie*, čiže tvorenie tvarov druhého a tretieho stupňa prídavných mien a prísloviek na vyjadrenie kvantity ich obsahu [1].

Ohýbaním jednotlivých slov sa vytvorí súbor prípon – vzor, ktorý sa opakuje pri viacerých slovách až sa vybuduje systém týchto súborov. Z týchto slov sa vyberie reprezentant vzoru, ktorý v najväčšej miere zastupuje daný vzor. Existujú vzory, ktoré zahŕňajú veľký počet slov (chlap, žena, kosť, vysvedčenie, ...), ale aj také, ktoré sa udržiavajú, kvôli veľkej frekvencii slov (gazdiná, idea). Nepravidelné vzory pozostávajú z 1-2 slov (deň, byť, ...). Sem patria aj slová, ktorých jedna časť sa skloňuje podľa jedného vzoru, druhá podľa iného, napr. zvieracie podstatné mená.

**Príklad 2.2:** Ohýbanie slov

**Skloňovanie**: dom

Singulár: dom, dom-u, dom-u, dom, dom-e, dom-om

Plurál: dom-y, dom-ov, dom-om, dom-y, dom-och, dom-ami

**Časovanie:** spať

prítomný čas: spím, spíš, spí, spíme, spíte, spia

budúci čas (byť + neurčitok): budem, budeš, bude, budeme, budete, budú spať

minulý čas: spal som, spal si, spal, spali sme, spali ste, spali

**Stupňovanie:** starý

starý, starší, najstarší

## 2.2 Tvorenie slov

Ako už bolo spomenuté jazyk je stále živý a vyvíja sa. Znamená to tiež, že sa modifikuje slovná zásoba jazyka. Miznú z nej slová, napríklad z dôvodu, že veci alebo predmety, ktoré pomenúvajú, stratili svoj význam a teda aj tieto pomenovania sa prestali používať. Zaujímavejšou časťou sú slová, ktoré naopak slovnú zásobu obohacujú. Rôzne spôsoby tvorenia slov sú rozobrané v nasledujúcich kapitolách.

### 2.2.1 Odvodzovanie slov

Pár pojmov na úvod [7]:

**slovotvorný základ:** časť odvodeného slova, ktorá je zhodná z pôvodným slovom alebo časťou pôvodného slova

**slovotvorná predpona**: časť odvodeného slova, ktorá stojí pred slovotvorným základom a mení jeho význam

**slovotvorná prípona**: časť odvodeného slova, ktorá stojí za slovotvorným základom a významovo ho obmieňa

Najčastejšie sa nové slovo odvodzovaní vytvára tak, že sa k slovotvornému základu pridá slovotvorný formant, teda slovotvorná predpona alebo prípona. Slovotvorným základom môže byť nielen slovo neodvodené (pôvodné), ale slovo už odvodené, zložené slovo prípadne predložkové spojenie.

**Príklad 2.3:** Odvodzovanie slov

| | | |
|---|---|---|
| Z neodvodených slov príponami: | | |
| učiť | → | učit-eľ |
| Z neodvodených slov predponami: | | |
| robiť | → | vy-robiť |
| Z odvodených slov: | | |
| učiteľ | → | učiteľ-stvo |
| Zo zložených slov: | | |
| malomesto | → | malomešť-an |
| Z predložkových spojení: | | |
| na brehu | → | ná-brež-ie |

Slová tiež môžeme odvodzovať konverziou, to znamená, že ostávajúcemu slovotvornému základu zmeníme súbor gramatických prípon. Často tak dochádza aj k zmene slovného druhu ako aj gramatických kategórii.

**Príklad 2.4:** Konverzia

| | | |
|---|---|---|
| zl-ý – prídavné meno<br>-ého, -ému, -om, ým, ... | → | zl-o – podstatné meno<br>-a, -u, -e, -om, ... |

Konverzia môže nastať aj v rámci slovného druhu:

**Príklad 2.5:** Konverzia v rámci slovného druhu

| | | |
|---|---|---|
| prút-0 – podstatné meno<br>-ého, -ému, -om, ým, ... | → | prút-ie – podstatné meno<br>-ia, -iu, -í, -ím, ... |

Špeciálnym prípadom je niekedy prechod z prídavného mena na podstatné meno bez zmeny súboru gramatických prípon. Jedná sa o takzvané spodstatnenie prídavného mena:

**Príklad 2.6:** Spodstatnenie prídavných mien

| | | |
|---|---|---|
| domáci – prídavné meno<br>chorý – prídavné meno | → | domáci – podstatné meno<br>chorý – podstatné meno |

Doteraz boli spomenuté len prípady konverzie ohybných slovných druhov, kde sa tvar slov väčšinou menil. Pri neohybných slovných druhoch je prechod slov medzi nimi jednoduchý, pretože sa neodráža na tvare slov. Niekedy sa preto nehovorí o prechode, ale o viacfunkčnosti slov [1].

**Príklad 2.7:** Viacfunkčnosť slov

okolo – príslovka: Išiel okolo, ale nevšimol si nás.
okolo – predložka: Obletel okolo stromu a pristál na skale.
okolo – častica: Zastavte sa okolo siedmej.

Slová je tiež možné tvoriť citovým zafarbením a to buď slová citovo zjemňujúce – zdrobneniny alebo slová citovo zhoršujúce – zveličené slová.

**Príklad 2.8:** Citové zafarbenie

strom – stromček: zdrobnenina
strom – stromisko: zveličené slovo

### 2.2.2 Skladanie slov

Skladaním slov vznikajú nové plnovýznamové zložené slová. Týmto spôsobom je možné vytvoriť podstatne menej slov ako odvodzovaním. Nové slová sa tvoria spojením dvoch slovotvorných základov a spájacej morfémy -o-, -e- alebo -i- [7].

V jazyku sa však vyskytujú aj takzvané nepravé zloženiny alebo zrazeniny. Formujú sa z dvoch slov bez spájacej morfémy.

Ďalším typom sú zložené slová, ktorých jedna časť cudzieho pôvodu. Sú časté najmä v odbornom jazyku.

V neposlednom rade treba spomenúť tvorenie zložených slov pomocou spojovníka.

**Príklad 2.9:** Skladanie slov

Pomocou spájacej morfémy:
zverolekár (zver-o-lekár), zemeguľa (zem-e-guľa), vlastizrada (vlast-i-zrada)
Zrazeniny:
kníhkupectvo, kníhtlač
Jedna časť cudzieho pôvodu:
ultraľahký, hyperrýchly, autoškola, elektromagnet
Skladanie pomocou spojovníka:
česko-slovenský slovník, bielo-modro-červená zástava

### 2.2.3 Skracovanie slov

Skracovanie slov slúži na úspornejšie vyjadrovanie predovšetkým v písomnej podobe. Poznáme nasledovné typy:

**Skratky** sa používajú len v písomnej podobe. Nepíše sa za nimi bodka a čítaju sa ako celé slová.

**Iniciálové značkové slová** sa vyskytujú v písomnom aj v ústnom prejave. Nepíše sa za nimi bodka, sú neohybné a nedajú sa od nich odvodzovať nové slová (iba hovorovo). Vznikajú mechanicky z prvých začiatočných písmen názvov všeobecne známych združení, celkov alebo inštitúcií. Vyslovujú sa ako samostatné slová abecedy.

**Značkové slová** sú tvorené zo začiatočných písmen alebo slabík viacslovných pomenovaní. Nepíše sa za nimi bodka. Podľa zakončenia sa zaraďujú k príslušným vzorom a skloňujú sa.

**Príklad 2.10:** Skloňovanie značkových slov

pomocou spojovníka: Tancujem v SĽUK-u

bez spojovníka: Čítal som o ufe. (UFO – Unidentified flying objects, neidentifikovaný lietajúci objekt)

Medzi **značky** patria napríklad medzinárodne dohodnuté označenia napríklad meracích jednotiek, chemických značiek, hudobných značiek alebo evidenčné čísla vozidiel

**Príklad 2.11:** Skracovanie slov

Skratky:

Dr. - doktor, napr. - napríklad, ved. - vedecký

Iniciálové značkové slová:

EÚ – Európska únia, SZĽH, - Slovenský zväz ľadového hokeja,

GJGT – Gymnázium Jozefa Gregora Tajovského

Značkové slová:

SĽUK – Slovenský ľudový umelecký kolektív, TANAP – Tatranský národný park,

SATUR – Slovenská turistická agentúra

Značky:

ms – milisekunda, kW – kilowatt (meracie jednotky)

He – hélium, H2SO4 – kyselina sírová (chemické značky)

f – forte, mf – mezoforte (hudobné značky)

BB 234CD – evidenčné číslo vozidla z banskobystrického okresu

Osobitným spôsobom skracovania slov je **univerbizácia**. Skrátením dvoj- a viacslovných pomenovaní vznikajú slová jednoslovné.

**Príklad 2.12:** Univerbizácia

minerálna voda – minerálka

panelový dom – panelák

Zaujímavým javom je skracovanie slov v písanej forme slangu na základe výslovnosti. Najlepším príkladom je anglický jazyk, kde je možné pomocou písmen a čísloviek nahradiť niektoré slová. Keďže v slovenčine sa samostatné písmená vyslovujú rovnako ako v rámci slov, je možné využiť len pár čísloviek, teda aj počet takýchto slov je nízky. Využitie má daný jav najmä na internete, v neformálnej elektronickej pošte a v SMS.

**Príklad 2.13:** Skracovanie v slangu

Anglické príklady:

C U SOON – See you soon. (Do skorého videnia.)

2 U – To you. (Pre teba.)

SK8 – skate (skateboard)

Slovenské príklady:

s5 – späť

o5 - opäť

# 3 Morfologický analyzátor

Pri morfologickej analýze slov v textoch, je potrebné si uvedomiť, že slová sa v texte nachádzajú v rozličných formách. Formy jedného slova je treba zjednotiť a vybrať najreprezentatívnejší tvar, ktorým je najčastejšie *lemma*. Tento proces sa nazýva *lemmatizácia* [3].

## 3.1 Porovnanie s inými jazykmi

Proces lematizácie sa svojou zložitosťou líši v rôznych jazykoch, záleží hlavne na počte foriem, ktoré slová tvoria ako aj od počtu nepravidelností. Spomedzi autorovi známych jazykov je tento proces pravdepodobne najľahší v anglickom jazyku, najmä kvôli malému počtu foriem slov. Toto využíva aj Porterov algoritmus [3], ktorý odstraňuje ustálené koncovky (napríklad *-s, -ed, -ing, -er, -est*) a teda dostáva lemma. Keď sa k tomuto algoritmu pridajú nepravidelnosti v stupňovaní prídavných mien a prísloviek (*good – better – best*), pri zámenách (*i – me, we – us, ...*) a pri tvorení minulého času a trpných príčastí slovies (*see – saw – seen, go – went – gone, ...*) mal by proces lemmatizácie pokryť viacmenej celú slovnú zásobu.

Zložitejší je prístup z pohľadu nemčiny, kde existuje rôzna sada koncoviek pre množné čísla podstatných mien a pády podstatných mien sú vyjadrené rôznymi členmi (*der, des, dem, den, die, das*) alebo koncovkami *-s* a *-n* v genitíve (*des Vaters* – (bez) otca), niekedy dokonca nastáva zmena v báze (*Väter* – otcovia). Koncovky sa pravidelným spôsobom menia pri skloňovaní prídavných mien a zámen (*-e, es, -em, ...*). Stupňovanie prídavných mien a prísloviek má podobne ako v angličtine niekoľko nepravidelných tvarov, pravidelné majú dokonca zhodné koncovky s anglickými (*-er, -est*), pri prídavných menách ešte doplnené o pádové prípony. Oproti anglickému jazyku sa rozšíril aj súbor prípon slovies, kde sa pri ich časovaní mení prípona v každej osobe prítomného aj minulého času (*springen* – skákať: *springe, sprigst, sprigt,...* – skáčem, skáčeš, skáče, *...; sprigte, springtest, springte, ...* – skákal som, skákal si, skákal). Okrem nepravidelných slovies tu niekedy nastáva aj zmena v tvarotvornom základe, napríklad a → ä, o → ö, u → ü. Ďalším problémom sú odlučiteľné predpony. Pri tvorení minulého príčastia sa vo väčšine prípadov používa predpona *ge-*. Avšak pri odlučiteľných príponách sa vkladá medzi túto príponu a slovo, čiže pri analýze slova, je potrebné najskôr odstrániť odlučiteľnú predponu, potom predponu *ge-*, získať základ slova a pripojiť odlučiteľnú predponu (zumachen – zatvoriť: zugemacht → gemacht → macht → machen → zumachen). Ešte komplikovanejšie je analyzovať slovesá s odlučiteľnými predponami vo vetách prítomnom a jednoduchom minulom čase, kedy tieto predpony stoja mimo slovies, najčastejšie na konci vety. Ak by sme teda chceli pracovať so skutočnými slovami, musel by sekvenčnému spracovaniu predchádzať vetný rozbor, ktorý by dokázal tieto predpony spojiť so slovesami.

Z románskych jazykov môžeme spomenúť španielčinu. Systém mien je pomerne jednoduchý. Prípony *-s, -es* sa používajú v množnom čísle a koncovky *-o, -a* rozlišujú rod mien. Príponami *-ito, -ita* je možné tvoriť zdrobneniny. Na stupňovanie sa okrem pár výnimiek využívajú neurčité číslovky *más* – viac a *el más* – najviac, čo na jednej strane uľahčuje určiť základný tvar, keďže je zhodný s tvarom stupňovaným, na druhej však neumožňuje priamo zo slova určiť jeho stupeň. Osobné zámená majú niekoľko tvarov: *yo* – ja, *mi* – mne, mňa, *conmigo* – so mnou, ... V priamom a nepriamom predmete sa vyskytujú ďalšie tvary, líšia sa v tretej osobe – *la, lo, las, los* (ju, ho, ich), *le, les* (jemu jej, im). Bez náväznosti na tvarotvorný základ (*se*) sa formujú zvratné zámená (*me, te, se, os, nos, se*). Základné číslovky sú nesklonné, radové sa správajú rovnako ako prídavné mená. Slovesá už podliehajú časovaniu, dokonca vykazujú značnú pravidelnosť, výtvarajú však rozsiahly súbor prípon. Slovesá sa v španielčine delia do troch skupín podľa koncoviek *-ar, -er, -ir* (*hablar* – hovoriť, *comer* – jesť, *vivir* – bývať). V každom čase, v ktorom sa nepoužíva pomocné sloveso, tvorí každá skupina súbor šiestich koncoviek. Ďalšími koncovkami *-ando, -iendo* sa tvorí príčastie prítomné, pomocou *-ado, -ido* príčastie minulé. Ako vidíme súbor prípon je naozaj veľký, ale jasne zaraďuje slovesá do gramatických tvarov. Náročnejší môže byť proces lemmatizácie, pretože niektoré prípony sú zdieľané aj medzi skupinami *-ar, -er,-ir*. Problém, ktorý by bolo treba riešiť nastáva v niektorých časoch slovies, kedy sa na sloveso viaže zvratné zámeno alebo zámeno pri priamom a nepriamom predmete (*dar* – dať: *Damelo!* - Daj mi to!, *casarse* – ženiť sa: *Voy a casarme.* - Idem sa ženiť.). Komplikácie môže spôsobiť aj akcent vyjadrený znakom „´“, ktorý sa pri ohýbaní slov, ak je potreba, dodatočne vyznačuje. Akcent by sa dal pri analýze zanedbať, pretože nevyjadruje gramatické javy.

Postavenie slov v anglických a nemeckých vetách je relatívne nemenné, tým pádom pozícia slova vete podáva podstatnú časť informácií o vzťahu k ostatným slovám vo vete [3]. V slovenskom jazyku (ako aj ostatných slovanských jazykoch) nie je postavenie slova dané pevne a môže sa meniť, a teda neurčuje jednoznačne jeho vzťah k ostatným slovám vo vete. Tento vzťah vyjadrujú jasnejšie koncovky slov. Slovenské slová majú obrovské množstvo tvarov, a teda aj koncoviek, ktoré sa používajú v rôznom kontexte. Ak by sme predpokladali, že slová vo vetách sa nachádzajú v slovníkovom tvare, úspešnosť ich nájdenia v slovníku by sa pravdepodobne blížila k nule. Z toho vyplýva, že prostriedky použité pri morfologickej analýze a lemmatizácii spomínaných jazykov sa nedajú použiť na slovenský jazyk.

## 3.2 Lemmatizácia

V slovenčine je prakticky nemožné vytvoriť súbor pravidiel podobných Porterovmu algoritmu slúžiacich na lemmatizáciu. Na Inštitúte informatiky Slovenskej akadémie vied a na Inšitúte

počítačových vied na Univerzite Pavla Jozefa Šafárika [3] navrhli prístup, založený na pozorovaní faktu, že ohýbanie slov závisí primárne na ich koncovkách.

Algoritmus potrebuje k dispozícii zoznam všetkých liemm a zoznam všetkých *vzorov* (vzorových vyskloňovaných liemm). Pre zadané slovo potom postupuje v troch fázach:

1. hľadá v zozname *vzorov* slová s rovnakou koncovkou
2. na základe vzťahu *vzoru* s jeho lemmou vytvorí všetky potenciálne lemmy zadaného slova
3. skontroluje, či sa kandidáti nachádzajú v zozname liemm

Ak je kandidátov viac, môže ich brať do úvahy všetky alebo len lemmu s najdlhšou spoločnou koncovkou. Celý algoritmus predpokladá, že časť slova, ktorá ostane po odstránení koncovky sa v procese lemmatizácie nemení. Postup algoritmu ukazuje nasledujúci príklad:

**Príklad 3.1:** Algoritmus lemmatizácie

Vstup: plavidiel

– slovo sa nenachádza v zozname *vzorov*
– jedno z nájdených *vzorov:* mydiel
– spoločná koncovka: -diel
– začiatok *vzoru*: my-
– začiatok zadaného slova: plavi-
– lemma *vzoru*: mydlo
– koncovka lemmy *vzoru*: -dlo
– pravdepodobná lemma zadaného slova: plavidlo
– kontrola slova v zozname liemm

Ak sa slovo nenachádza v zozname *vzorov* a nachádza v zozname liemm, znamenáto, že ho treba vyskloňovať (vyčasovať) a spolu so všetkými formami uložiť do zoznamu vzorov.

Nie vždy je však možné popísať tvorenie tvarov pomocou jednoduchých pravidiel na základe koncoviek. V slovenčine sa veľmi často vyskytujú zmeny v báze, najmä v genitíve množného čísla. Teoreticky je možné zo všetkých slov so zmenou v báze vytvoriť všetky tvary a uložiť do zoznamu vzorov, čo zaberie veľa času. Ďalšou možnosťou je nájdenie pravidiel, ktoré by tieto zmeny popisovali. Dôvodom zmeny v báze je dodržovanie zákona o rytmickom krátení, ktorý hovorí, že dve po sebe nasledujúce slabiky nemôžu byť dlhé. Tvorenie genitívov v množnom čísle od liemm ženského rodu končiacich na *-a* môže byť popísané nasledovne [3]:

- ak sa končí na *-ia,* genitív plurálu sa bude končiť na *-ií* (línia → línií)
- ak sa končí na *-ka*, pred ktorou je spoluhláska
  - ak je predposledná slabika dlhá, nahradí sa *-ka* za *-ok* (láska → lások)
  - ak je predposledná slabika krátka, nahradí sa *-ka* za *-iek* (letka → letiek)
- ak sa končí na *-ka*, pred ktorou je samohláska
  - ak je predposledná slabika dlhá, odstráni sa *-a* (rieka → riek)
  - ak je predposledná slabika krátka a slabika pred ňou je dlhá, tiež sa odstráni -a (bábika → bábik)
  - ak je predposledná slabika krátka a slabika pred ňou je tiež krátka alebo neexistuje, potom sa predposledná slabika nahradí jej dlhým variantom podľa pravidiel v tabuľke 3.1 a koncové *-a* sa odstráni
- ak sa končí na *-a,* pred ktorým spoluhláska (rôzna od *k*), dôjde k zmene k zmene predposlednej slabiky podľa pravidiel v tabuľke 3.1 koncové *-a* sa odstráni

**Tabuľka 3.1:** Pravidlá zmien krátkych slabík na ich dlhé varianty

| | | | |
|---|---|---|---|
| a → ia | vďaka → vdiak, ťarcha → tiarch | o → ô | stoka → stôk, bomba → bômb |
| a → á | straka → strák, sada → sád | u → ú | ponuka → ponúk, huba → húb |
| ä → ia | mäta → miat | y → ý | motyka → motýk, rakyta → rakýt |
| e → ie | paseka → pasiek, pena → pien | l → Í | odmlka → odmÍk, vlna → vÍn |
| i → í | panika → paník, koliba → kolíb | r → ŕ | sprcha → spŕch |

Pre samohlásku *a* existujú dve pravidlá. O tom, ktoré vybrať, rozhoduje spoluhláska stojaca pred *a.* Ak je mäkká, použije sa pravidlo a → ia, inak sa použije pravidlo a → á. Spoluhlásky *l, r, Í, ŕ* sú slabikotvorné. Znamená to, že ak sa nachádzajú medzi dvoma spoluhláskami správajú sa pri tvorení slabík ako samohlásky (vl-na, vŕ-ba) a horeuvedené pravidlá platia len pre tento prípad.

Pravidlá pre slová končiace sa na *-a* (okrem *-ka*) nie sú také striktné a obsahujú ďalšie výnimky, ktoré by bolo treba popísať (marža → marží, karta → karát/kariet).

Podobné pravidlá by sme mohli vytvoriť aj pre lemmy stredného rodu končiace na *-o* (jablko → jabĺk, oko → ôk, pravidlo → pravidiel, cesto → ciest, autorstvo → autorstiev, ...).

Ak sa pri lemmatizácii nenájde tvar slova medzi zoznamom *vzorov*, použije sa opačný postup ako bol popísaný pri tvorení genitívov množných čísiel ženského rodu, čím z daného tvaru dostaneme lemma. Tvary, od ktorých sa ani potom nepodarí vytvoriť lemma, môžu slúžiť k vytváraniu nových pravidiel alebo ak sa jedná o výnimku, k vytvoreniu tvarov a pridaniu do zoznamu *vzorov*.

Podobný prístup k lemmatizácii bol vyvinutý Janom Daciukom z univerzity v Gdaňsku, na ktorý nadviazali aj pracovníci v Česku. Tento prístup využíva rozsiahlych slovníkov so všetkými tvarmi, pri ktorých je zakódovaná lemma nasledujúcim spôsobom pomocou Kending (príklad 3.2). Kending sa vytvára z dvoch slov a skladá sa z veľkého písmena abecedy a koncovky. Veľké písmeno abecedy udáva, koľko znakov sa má z prvého slova odobrať, aby po pripojení koncovky z druhé slova vzniklo druhé zadané slovo. Veľké A znamená, že sa neodtrhne žiaden znak, B jeden znak, C dva znaky, atď. Koncovku určíme tak, že od druhého zadaného slova oddelíme spoločnú časť oboch slov.

**Príklad 3.2:** Kendings

*mamám+mama*
uloženie tvaru:
mama+Bám
uloženie lemmy:
mamám+Ca

Proces lemmatizácie je potom nasledovný. Najskôr sa tvar zadaného slova vyhľadá v slovníku. Od jeho konca sa odstráni počet písmen daný veľkým písmenom Kending. K vzniknutému základu sa „prilepí“ koncovka z Kending a vzniká lemma. Pri nájdení slova v slovníku má proces 100% úspešnosť a presnosť a je možné mať uložené viaceré lemmy k rovnakým tvarom. Pre slová, ktoré sa v slovníku nenachádzajú je použitý podobný postup ako bol popísaný vyššie. Ako je vidno na príklade 3.2, môžeme využiť aj opačný postup, a to tvorenie všetkých tvarov od istej lemmy. Problém s vyššie spomínanými podstatnými menami ženského a stredného rodu odpadá, pretože zmena v báze sa prejaví len v zvýšení počtu znakov koncovky. Iné riešenie by bolo, pridať značku za Kending informujúcu o zmene v báze s popisom tejto zmeny. Teoreticky by sa tým dala znížiť veľkosť slovníka. Veľkosť slovníka by mohla byť záťažou pre takýto spôsob lemmatizácie. Ako sa tento problém rieši, je popísané v nasledujúcej kapitole.

# 4 Uloženie dát

Podľa informácii v práci Pavla Šmerka [2], existujú pri návrhu morfologického analyzátora flektívnych jazykov (jazyky, ktorých slová sa ohýbajú) dve možnosti uloženia dát:

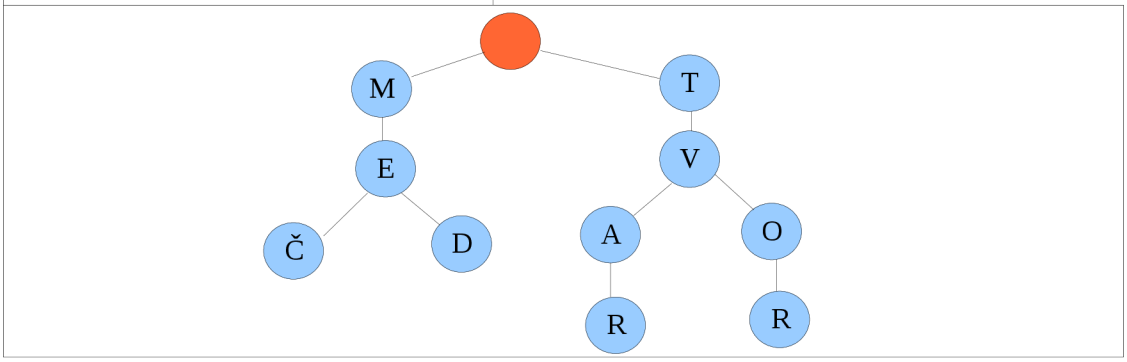
- vygenerovanie všetkých gramatických tvarov a vytvorenie systému s veľkým slovníkom a veľmi jednoduchou analýzou
- vybudovanie systému s menším slovníkom koreňov s informáciou o možných koncovkách, ale použitím sofistikovanejšieho algoritmu

Oficiálny slovenský slovník, Slovník slovenského jazyka, obsahuje 150 000 liemm, čo predstavuje približne 5 miliónov tvarov [3]. Keď pridáme ku každému tvaru gramatické informácie, vzor, poprípade poznámku, pamäťové nároky viditeľne vzrastú. Určite je to jeden z dôvodov, prečo je v horeuvedenom dokumente odporúčaný druhý variant. Ďalšou nevýhodou je sekvenčné vyhľadávanie na veľkom objeme dát, ktoré by spomaľovalo analýzu. Čiastočným riešením by mohlo byť rozdelenie slovníka na základe začiatočných písmen.

Čo ak ale uložíme slovník v inej ako textovej podobe? Vhodnou dátovou štruktúrou na uloženie v binárnej forme je TRIE. Je to n-nárny vyhľadávací strom. Na rozdiel od binárneho vyhľadávacieho stromu, ktorý priraďuje každému uzlu jeden kľúč, ukladá reťazec kľúča tak, že každému uzlu priradí jeden znak kľúča. Nazýva sa tiež prefixový strom, pretože jednotlivé kľúče zdieľajú svoje prefixy s kľúčmi s rovnakým prefixom [4].

Na obrázku 4.1 je zobrazená dátová štruktúra TRIE, v ktorej sú uložené kľúče MEČ, MED, TVAR a TVOR. Kľúče MEČ a MED zdieľajú spoločný prefix ME a kľúče TVAR a TVOR majú spoločný prefix TV. Pri ohýbaní slov sa menia len prípony tvarotvorný základ ostáva. Z toho vyplýva, že pri uložení všetkých tvarov jednej lemmy budú všetky tvary zdielať tvarotvorný základ.

**Obrázok 4.1:** Dátová štruktúra TRIE



Ako je možné vidieť na obrázku 4.1, dátová štruktúra TRIE je podobná konečným automatom. To nabáda k využitiu minimalizačných techník používaných pri minimalizácii konečných automatov.

## 4.1 Acyklický deterministický konečný automat

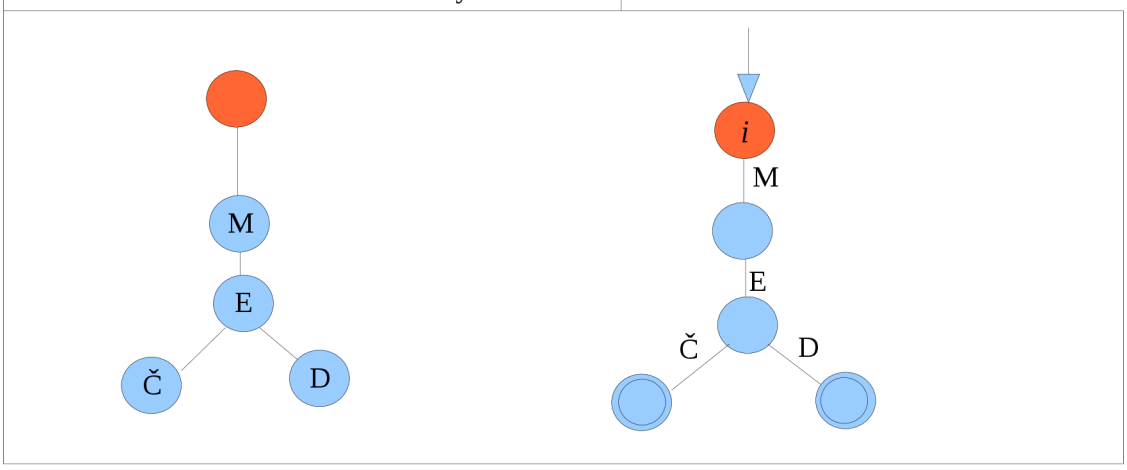
Dokument Jana Daciuka [5] definuje acyklický deterministický konečný automat ako päticu $A = (\Sigma, Q, i, F, E)$, kde $Q$ je konečná množina stavov $i \in Q$ je počiatočný stav, $\Sigma$ je vstupná abeceda $F \subseteq Q$ je množina koncových stavov a $E \subseteq Q \times \Sigma \times Q$ je množina prechodov.

Acyklický znamená, že neobsahuje žiadne cykly, teda z aktuálneho stavu môže prejsť len do stavu, v ktorom pri spracovaní daného reťazca nebol, mimo stavy, z ktorých sa do stavu v ktorom bol môže dostať.

Deterministický automat neobsahuje $\varepsilon$ prechody, nedovoľuje prechod medzi stavmi bez načítania znaku z reťazca. Determinizmus tiež vylučuje, aby sa po načítaní znaku z reťazca dalo prejsť z jedného stavu do viacerých stavov.

TRIE na konečný automat prevedieme jednoduchým algoritmom. Koreňový uzol štruktúry označíme za počiatočný stav konečného automatu. Hodnoty uzlov priradíme hranám jednotlivých prechodov. Nakoniec označíme koncové stavy, teda stavy, z ktorých nevedú žiadne prechody [4] (Obrázok 4.2) .

**Obrázok 4.2** Prevod TRIE na konečný automat

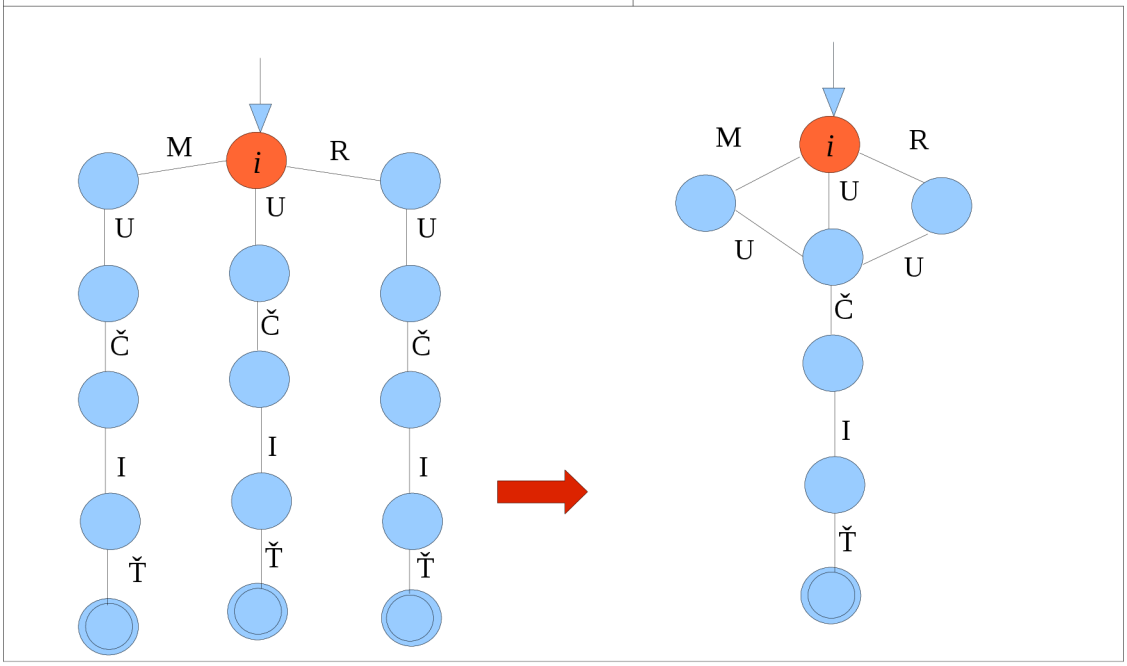


## 4.2 Minimalizácia konečných automatov

Použitím konečných automatov sa nám teda naskytla možnosť tieto automaty minimalizovať. Základná myšlienka je vysvetlená na obrázku 4.3. Majme slová UČIŤ, RUČIŤ a MUČIŤ. Vidíme, že všetky majú spoločnú koncovú časť UČIŤ. Túto spoločnú časť stačí teda v automate uložiť len raz.

Pri skloňovaní sa pre rovnaké gramatické kategórie (rod, číslo, pád) opakujú tvarotvorné prípony a vznikajú vzory. Každá tvarotvorná prípona sa teda podobne ako to bolo v predchádzajúcej ukážke uloží len raz. Podobné vzory vnikajú aj pri stupňovaní a časovaní.

**Obrázok 4.3:** Minimalizácia konečných automatov




**Príklad 4.1:** Vzory tvarotvorných prípon

Skloňovanie:

mesto: -o, -a, -u, -e, -om, á, -ám, -ách, -ami

mäso: -o, -a, -u, -e, -om, á, -ám, -ách, -ami

Stupňovanie:

mladý → mladší → najmladší: -ý, -ší, -ší

nový → novší → najnovší: -ý, -ší, -ší

Časovanie:

pracovať: -ujem, -uješ, -uje, -ujeme, -ujete, -ujú

cestovať:-ujem, -uješ, -uje, -ujeme, -ujete, -ujú

# 5 Príprava dát

Slovníkové dáta sú dôležitou súčasťou celého projektu. Ich forma a uloženie sa opiera o informácie spomenuté v kapitole 4. Slovenské dáta vychádzajú zo slovníka uloženého vo formáte uvedonenom v nasledujúcom príklade :

**Príklad 5.1:** Formát slovenských dát

tvar <l>lemma <c>tag
kde tag predstavuje gramatické kategórie

Tento tvar prevedieme jednoduchým algoritmom na podobný formát, kde sú jednotlivé položky oddelené oddeľovačom '+', aby sme mohli s dátami ľahšie pracovať. Ďalej je potrebné pripojiť ku každej forme Kending (príklad 3.2). Vytvoríme teda slovníky typu *lemma+Kending+tag*, z ktorých vieme získať všetky tvary liemm a slovníky typu *tvar+Kending+tag*, z ktorých vieme získať lemmy. V predchádzajúcej práci Stanislava Černého [4], ma ktorú na táto práca nadväzuje a rozširuje ju, bol k týmto zápisom v slovníku pridaný vzor. Nejedná sa však o gramatický vzor (chlap, hrdina, dub, stroj, ...), ale o technický vzor, ktorý združuje slová s rovnakým systémom tvarotvorných prípon.

## 5.1 Systém vzorov

Aby sme systém vzorov mohli vytvoriť, potrebujeme systém prípon. Ten vytvoríme tak, že vezmeme slovník formátu *lemma+Kending+tag* a na jeden riadok nového súboru vypíšeme všetky *Kendings* jednej lemmy. Tento súbor ešte preženieme cez skript `uniq` a dostaneme súbor všetkých možných systémov prípon. Ďalší súbor, ktorý potrebujeme, vytvoríme podobným spôsobom ako ten predchádzajúci, s tým rozdielom, že už nepoužijeme skript `uniq` a ku každému riadku pripojíme lemmu oddelenú znakom '#', ku ktorej sa *Kendings* vzťahujú. Máme teda dva súbory v nasledujúcom formáte:

**Príklad 5.2:** Formát súborov na získanie vzorov

Riadok zo súbor všetkých možných prípon (súbor 1):
*Bti Btí Btiam Bti Bti Btiach Aami A Bti Bti A A Bti Aou*
Riadok zo súboru *Kendings* #lemma (súbor 2):
*Ay Aom Ay Ay Aoch Ami A Aa Au Au A A Ae Aom #zrub*

Potom prechádzame súborom 2 a porovnávame *Kendigs* (reťazec po znak '#') s riadkami v súbore 1. Pri zhode zapíšeme lemmu zo súboru 2 do zoznamu k danému reťazcu *Kendings*. Po prejdení celého súboru 2 by sme mali dostať v podstate súbor 1 s pridanými lemmami k jednotlivým *Kendings*.

Z vytvorených zoznamov liemm pre jednotlivé *Kendings* je potrebné vybrať najreprezentatívnejšiu lemmu, ktorá bude tento vzor *Kendings* reprezentovať . Reprezentant vzoru musí spĺňať niekoľko podmienok:

- musí všetkými tvarmi patriť do systému vzorov, nesmie byť „vínimkou pravidla“
- nesmie tvoriť zložené slová (pán – mocipán, mesto – veľkomesto)
- a tiež nesmie byť zloženým slovom

Tieto podmienky sa dajú striktne dodržať len dodatočnou analýzou každého slova. Jednoduchšou a schodnejšou cestou, ale zároveň nájdením relatívne vhodného reprezentanta, je použitie frekvenčného slovníka. Vo frekvenčnom slovníku je ku každému slovu priradený počet výskytov slova v textoch všeobecne. Tento počet bol získaný z veľkého objemu dát v projekte NLP. Za reprezentanta vzoru vyberieme tú lemmu, ktorej počet výskytov je najväčší spomedzi všetkých liemm v zozname liemm pri danom vzore *Kendings*.

Teoreticky je možné využiť systém gramatických vzorov (chlap, hrdina, dub, stroj, ...) [6], ale ako bolo spomenuté v kapitole 3, v slovenskom jazyku sa nachádza veľa nepravidelností, ktoré tomu zabraňujú. Pokrytie podstatných mien gramatickými vzormi je znázornené v nasledujúcej tabuľke:

**Tabuľka 5.1:** Pokrytie podstatných mien gramatickými vzormi

| |
|---|
| **Počet podstatných mien**: 24 631 |
| chlap: 1297, hrdina: 96, dub: 231, stroj: 372, kuli: 5, |
| žena: 18, ulica: 429, dlaň: 15, kosť: 2 817, gazdiná: 4, idea: 6, |
| mesto: 5, srdce: 2, vysvedčenie: 1 023, dievča: 29 |
| nezaradené: 18 282 |
| |
| Percentuálne pokrytie: 25,78% |

Z výsledkov vyplýva, že presné pokrytie gramatickými vzormi len jednu štvrtinu podstatných mien v slovníku. Najväčším problémom je rytmické krátenie, prechyľovanie kmeňových hlások a zmäkčovanie, čo sa najviac prejavuje v druhom páde množného čísla. Druhý experiment ukazuje výsledky bez týchto tvarov.

**Tabuľka 5.2:** Pokrytie podstatných mien gramatickými vzormi bez genitívu plurálu

**Počet podstatných mien**: 24 631

chlap: 1297, hrdina: 96, dub: 231, stroj: 372, kuli: 5,

žena: 4549, ulica: 520, dlaň: 15, kosť: 2 817, gazdiná: 11, idea: 6,

mesto: 743, srdce: 13, vysvedčenie: 1 023, dievča: 29

nezaradené: 12 885

Percentuálne pokrytie: 47,69%

Gramatické vzory bez genitívu pokryjú takmer polovicu podstatných mien v slovníku. Tento tvar nie je rozhodujúci pri určovaní technických vzorov a môžeme ho zanedbať, čím dostaneme lepšie pokrytie a menej technických vzorov (tabuľka 5.3).

**Tabuľka 5.3:** Technické vzory podstatných mien

Technické vzory: 785

Bez genitívu plurálu: 417

V ďalších slovných druhoch už nie sú výnimky také časté takže technické vzory môžeme doplniť do dát(tabuľka 5.4). Pri neohybných slovných druhoch dostaneme týmto spôsobom len minimálny počet technických vzorov pre celý slovný druh: Niektoré príslovky môžeme stupňovať a niektoré nie, pričom niektoré sa stupňujú nepravidelne, čo dá za výsledok niekoľko vzorov. Na ďalšie triedenie vzorov by sme ako kritérium mohli použiť druh prísloviek (času, miesta, spôsobu a príčiny). Pri spojkách sa tiež objavujú rôzne tvary, napríklad pri vokalizovaných predložkách (v, vo, k, ku) a tiež rôzne predložky sa môžu viazať z rôznym počtom pádom, čo tiež ovplyvní systém vzorov. V ostatných prípadoch vznikne popísaným spôsobom len jeden technický vzor.

**Tabuľka 5.4:** Technické vzory celkovo

| | |
|---|---|
| Podstatné mená: 417 | Predložky: 9 |
| Prídavné mená: 38 | Spojky: 1 |
| Zámená: 54 | Častice: 1 |
| Číslovky: 30 | Citoslovcia: 1 |
| Slovesá: 317 | Skratky: 1 |
| Príslovky: 25 | Podmienky (by): 1 |

Celkovo: 895

## 5.2 Predpony

Predpony budú uložené spolu s ostatnými slovami v automate, ale aby ich analyzátor nepovažoval za slová budú od kategórií oddelené znakom '!'. Ak sa pri analýze zistí, že zatiaľ načítaná časť slova je predpona, pokračuje sa v analýze zvyšnej časti slova od začiatku. Takýmto spôsobom je možné analyzovať viac predpôn v jednom slove.

Nie vždy je žiadúce, aby bola prepona pripojená k slovu. Takéto výnimky sú ošetrené tak, že pred kategórie daného slova pridáme znak '!'. Predpony, ktoré nechceme zahŕňať k základným tvarom slov uložíme so špeciálnou značkou '#' medzi výkričníkom a kategóriami.

Podrobnejšie informácie o uložení predpôn sa nachádzajú v dokumente [4].

## 5.3 Balík *fsa*

Poslednou fázou pri príprave dát je ich transformácia z textovej podoby na binárnu. K tomu poslúži balík *fsa*[1], ktorý vytvoril Jan Daciuk. Balík obsahuje celú sadu programov vhodných k prirodzenému spracovaniu jazyka ako napríklad *spellchecker* na kontrolu pravopisu, program na doplnenie diakritiky, program plniaci úlohu morfologického analyzátora, program na približnú morfologickú analýzu na základe prípon a predpôn a čo je najdôležitejšie, program *fsa_build,* ktorý z našich dát vytvorí konečné automaty.

Program *fsa_build* očakáva na vstupe formát dát: *form+Kending+annotation.* Anotácia pôvodne predstavovala gramatické kategórie, ku ktorým sme pridali vzor. To by na základe pravidiel minimalizácie konečných automatov (kapitola 3) malo veľkosť výsledného automatu ovplyvniť len minimálne, vzhľadom na fakt, že ten istý vzor sa pridá k rovnakým koncovkám a počet vzorov je niekoľkonásobne nižší ako počet tvarov.

1 Viac informácií, kompletný popis a balík fsa k stiahnutiu na http://www.eti.pg.gda.pl/katedry/kiw/pracownicy/Jan.Daciuk/personal/fsa.html#FSApack

# 6 Implementácia

Na implementáciu bola použitá knižnica *libma*, ktorá využíva spomínaný balík *fsa*. Je napísaná v objektovo orientovanom jazyku C++, takže všetky jej rozšírenia ako aj rozhranie k tejto knižnici je napísané v tomto programovacom jazyku. Súčasťou implementácie sú teda rozšírenia knižnice *libma* a jej rozhrania *ma*. Knižnica ponúka niekoľko dátových štruktúr na uloženie výsledkov vyhľadávania v slovníkoch. Líšia sa najmä veľkosťou jednotlivých položiek, čo sa potom odráža aj na rýchlosti spracovania dát.

Hlavným pozitívom knižnice je zapúzdrenie týchto dátových štruktúr do objektov, tým pádom sú všetky procesy ohľadom načítavania a spracovania dát skryté pri budovaní a používaní rozhrania. Nevýhodou oproti balíku *fsa* je práve toto zapúzdrenie. Balík *fsa* využíva vlastné optimalizované typy, ktoré sa v knižnici transformujú na typy z *STL* [4].

Okrem štruktúr obsiahnutých v knižnici, ktoré slúžia na uloženie dát pri hľadaní základného tvaru, a štruktúr, do ktorých sa ukladajú dáta pri tvorení všetkých tvarov boli pridané nové štruktúry. Jedna slúži na rýchle hľadanie všetkých tvarov a používa slovník bez gramatických kategórií. Aby však bol výstup funkcie na tvorbu všetkých tvarov čo najviac konfigurovateľný, je potrebné použiť slovník aj s gramatickými kategóriami, čo sa ovplyvní rýchlosť spracovania. Pre túto funkciu bola tiež vytvorená špeciálna štruktúra.

Základom správneho a optimálneho chodu programu však ostávajú dobre pripravené rozsiahle dáta. Rozhodujúcim faktom je aj správne uloženie týchto dát popísané v kapitole 5.

## 6.1 Rozhrania *ma*

Rozhranie *ma* je textovým rozhraním knižnice *libma* dostupné a spustiteľné v príkazovom riadku. Jeho súčasťou je niekoľko funkcií popísaných v návode, ktorý sa zobrazí po spustení s parametrom -h (viď Príloha 1). Program očakáva slová oddelené „bielymi“ znakmi na štandardnom vstupe, výsledok spracovania slov vytlačí na štandardný výstup.

Hlavnou funkciou programu je určenie základného tvaru a gramatických kategórií zadaného tvaru slova na základe princípov popísaných v kapitole 3. Pri nájdení slova v slovníku, program pokračuje v hľadaní ďalších tvarov, ktoré sa líšia v gramatických kategóriach alebo vzoroch alebo v oboch spomínaných prípadoch. Ďalšia funkčnosť je spojená s generovaním všetkých tvarov, a to nie len od základného tvaru, ale od akéhokoľvek zadaného tvaru, tým spôsobom, že najskôr sa od zadaného tvaru získa tvar základný. Jednotlivé funkcie generujúce všetky tvary sa od seba v rýchlosti spracovania, vo formáte zobrazovaných dát a v konfigurovateľnosti výstupu. V prvej verzii sa generujú len tvary, bez akýchkoľvek ďalších informácií. V druhej verzii sú dáta zobrazované

s gramatickými kategóriami a v tretej, zatiaľ poslednej verzii, sú síce dáta zobrazované bez gramatických kategórií, ale na základe gramatických kategórií je výstupné dáta možné radiť alebo filtrovať.

Zaujímavým doplnkom rozhrania je výstup v takzvanom prezentačnom móde. Ako už názov napovedá, využitie nájde najmä pri prezentovaní programu a jeho úlohou je zobrazovať dáta pre užívateľa – laika zaujímavom formáte v zarovnaných stĺpcoch, čo ide opäť na úkor výkonu.

## 6.2 Radenie a filtrovanie

Jednou z požiadaviek na rozhranie bol jednoducho konfigurovateľný výstup. Znamená to, že užívateľ by mal vedieť jednoduchým spôsob zobraziť len dáta, ktoré potrebuje a v takom formáte ako potrebuje.

Vhodným spôsobom by bolo zadávanie kľúčov, podľa ktorých sa má radiť a filtrovať, do konfiguračného súboru (viď Príloha 2), z ktorého sa tieto informácie pri spustení programu načítajú a ďalej spracujú pomocou konečných automatov.

Radenie je zabezpečené zabudovanou funkciou *sort*(), ktorá okrem iného umožňuje radiť na základe vlastnej porovnávacej funkcie. Táto možnosť je samozrejme využitá a porovnávacia funkcia na základe jednotlivých položiek kľúča určuje poradie slov vo výsledku. Kľúč, na základe ktorého sa radí, môže obsahovať skratky gramatických kategórií alebo skratky ďalších kritérií radenia (abecedne podľa tvaru, podľa vzoru, ...). Skratky sú popísané v Prílohe 2.

Filtrovanie je podobne ako radenie vykonávané na základe zadaného kľúča, ktorý v tomto prípade obsahuje len gramatické kategórie, ku ktorým je pridaná hodnota gramatickej kategórie. Vytvorené sú dva filtre: filter „in“ alebo pozitívny filter a filter „out“ alebo negatívny filter. V prvom prípade sú gramatické kategórie porovnávané s hodnotami gramatických kategórií filtra a len v prípade zhody sa tieto slová zobrazia vo výsledku. Druhý filter naopak v prípade zhody dané slovo nezobrazí. Druhý filter využijeme napríklad vtedy, ak chceme zobraziť len niektoré pády. Ak by sme na to použili prvý filter, nezobrazilo by sa nám nič, pretože žiadna gramatická kategória nemôže obsahovať viac pádov. Keď však do druhého filtra zadáme doplnok k pádom, ktoré chceme zobraziť (teda tie, ktoré zobraziť necheme), dostaneme požadovaný výsledok.

**Príklad 6.1:** Porovnanie funkcie filtrov pre slovo *mesto*

| **Kľúč filtra** | **Výsledok** |
|---|---|
| Filter in: *c1c2c3* | *Not found* |
| Filter out: *c4c5c6c7* | mesto, mesta, mestu, mestá, miest, mestám |

## 6.3 Diakritika

Dôležitou vlastnosťou rozhraní by mala byť jednoduchosť a užívateľská prístupnosť. Viaceré jazyky ako aj ten slovenský obsahujú znaky, ktoré sa bežne nevyskytujú na „anglickej klávesnici“. Jedná sa o znaky, ktoré obsahujú dĺžeň, mäkčeň, dvojbodku, tildu a iné, ktoré sa súhrnne nazývajú diakritické znamienka. Často tieto znaky zadávame z rôznych dôvodov bez týchto diakritických znamienok a používaný systém nám zadané reťazce vyhodnotí, ako keby boli s týmito znamienkami zadané. Jedna z možných implementácií dopĺňania diakritiky je popísaná v nasledujúcom algoritme.

**Algoritmus 6.1:** Dopĺňanie diakritiky

1. Inicializuj zoznam kombinácií, do ktorého sa budú ukladať nové slová s informáciou(číslom), od ktorej pozície treba začať dopĺňať diakritiku.
2. Inicializuj zoznam nových slov.
3. Slovo, ku ktorému treba pridať diakritiku, ulož do zoznamu nových kombinácií s číslom nula.
4. Ulož ho do zoznamu nových slov.
5. **Kým** je nejaké slovo v zozname kombinácií tak
6. **Pre** každé slovo v zozname kombinácií
7. **Ak** sa aktuálne písmeno nachádza v akcentačnej tabuľke
8. **Pre** každé písmeno od danej pozície
9. Priraď akcenty z akcentačnej tabuľky daného slova
10. Ulož nové slová do zoznamu nových slov.
11. **Ak** sa nejedná o posledné písmeno
12. Ulož vytvorené slová do zoznamu kombinácií.
13. Vymaž pôvodné slovo zo zoznamu kombinácií.
14. **Inak**
15. **Ak** sa nejedná o posledné písmeno
16. Zvýš pozíciu písmena tohto slova v kombináciách
17. **Inak**
18. Vymaž slovo zo zoznamu kombinácií.

Základom algoritmu je akcentačná tabuľka, ktorá obsahuje dva stĺpce. V prvom sú všetky znaky, ktoré sa objavujú v jazyku s diakritickými znamienkami a v druhom sú reťazce znakov s týmito znamienkami (viď Príloha 3). Táto tabuľka je uložená v súbore s akcentami, kde sú jednotlivé stĺpce oddelené medzerou. Aby sa dal tento súbor načítať, bola knižnica *libma* rozšírená

o nový modul, ktorý celé načítanie zabezpečuje. Modul tiež obsahuje metódu, ktorá využíva popísaný algoritmus k tvorbe zoznamu slov s diakritickými znamienkami. Akým spôsobom algoritmus spracúva slová je ukázaný v nasledujúcom príklade.

**Príklad 6.2:** Pridanie diakritiky slovu *hada*

1. Kombinácie: hada – 0
   ***h***ada – písmeno nie je v akcentačnej tabuľke, zvýš pozíciu
2. Kombinácie: hada - 1
   h***a***da – doplní *á* a *ä* – uloží ku kombináciam a k novým slovám
   ha***d***a – doplní *ď* – uloží ku kombináciam a k novým slovám
   had***a*** – doplní *á* a *ä* – posledné písmeno, slovo uloží len k novým slovám
3. Kombinácie: háda – 2, häda – 2, haďa - 3
   há***d***a, hä***d***a – doplní *ď* – uloží ku kombináciam a k novým slovám
   hád***a***, häd***a*** – doplní *á a ä* – posledné písmeno, slovo uloží len k novým slovám
   haď***a*** – doplní *á a ä* – posledné písmeno, slovo uloží len k novým slovám
4. Kombinácie: háďa – 3, häďa – 3
   háď***a***, häď***a*** – doplní *á* a *ä* – posledné písmeno, slovo uloží len k novým slovám

**Obsah výsledného zoznamu nových slov**:
hada, háda, häda, haďa, hadá, hadä, háďa, häďa, hádá, hádä, hädá, hädä, haďá, haďä, háďá háďä, häďá, häďä

Algoritmus na pridávanie diakritiky vytvorí všetky možné slová s diakritikou aj keď väčšina nemá zmysel. Zmenšiť počet vygenerovaných slov by sa podarilo pridaním jednoduchej logiky. Napríklad široké ä nasleduje v slovenčine len po spoluhláskach *b, m, p, v,* na spoluhláskach *d, t, n, l* nemôže byť mäkčeň ak po nich nasleduje samohláska *e* alebo *i*, ktorá tieto spoluhlásky zmäkčuje. Zložitejším spôsobom by sa dal dodržovať zákon o rytmickom krátení, ktorý hovorí, že dve po sebe idúce slabiky nemôžu byť dlhé. Tento zákon však obsahuje veľa výnimiek, napríklad zvieracie prídavné mená podľa vzoru páví, či podstatné mená so slabikami obsahujúcimi slabikotvorné *Ĺ* a *ŕ* – *tŕnie (v tŕní)*. Jednoduchším vylepšením algoritmu je použitie frekvenčného slovníka. Z vygenerovaných slov sa vyberie slovo, ktoré sa vo všebecnosti vyskytuje najčastejšie a toto slovo sa potom ďalej analyzuje.

## 6.4 Testy a analýzy

Súčasťou práce je vyhodnotenie časových a pamäťových nárokov aplikácie. Na to je potrebné vykonať sériu niekoľkých testov a pozorovaní. Všetky ďalej uvádzané testy boli vykonané na stroji s parametrami: Intel(R) Core(TM)2 Quad CPU Q6700 @ 2.66GHz, RAM 4GB, CentOS release 5.3 (Final), GNU/Linux kernel release 2.6.27.21.

### 6.4.1 Pamäťové nároky

Ako už bolo spomenuté program využíva rozsiahly slovník, ktorý obsahuje ku každému základnému tvaru vygenerované všetky jeho tvary. K dispozícii bol v čase implementácie slovník s 59 171 základnými tvarmi, čo spolu s vygenerovanými tvarmi činí 1 762 523 slov. Nesmieme zabudnúť, že ku každému slovu je priradený *Kending* (príklad 3.2), značky s gramatickými kategóriami a vzor. Na toto množstvo dát sme použili program *fsa_build*, ktorý z nich vytvoril minimalizovaný konečný automat. K ďalšiemu zmenšeniu prispelo zakódovanie predpôn do dát, tým pádom mohli byť všetky negatívne tvary a všetky superlatívy vymazané. Lepšiu predstavu o veľkosti zaberajúceho priestoru poskytne nasledujúce tabuľka.

**Tabuľka 6.1:** Porovnanie pamäťových nárokov slovníkov v textovej a binárnej forme

| | **s predponami** | **predpony zakódované v dátach** |
|---|---|---|
| **textová forma** [kB] | 60 865 | 60 774 |
| **konečný automat** [kB] | 1 597 | 859 |

V tabuľke 6.1 vidno, že zakódovanie predpôn nespôsobilo až taký rozdiel oproti pôvodnej verzii v textovej podobe. Je to ale spôsobené tým, že v pôvodnom slovníku sa nenachádzajú superlatívy prídavných mien a prísloviek a negované tvary slovies, ktoré sme zakódovaním predpôn doplnili. Napriek tomu je však výsledný konečný automat o polovicu menší ako pôvodný.

Vhodné by bolo porovnať slovenský a český slovník, ktorý obsahuje približne 745 329 základných tvarov a spolu je všetkých tvarov 40 843 228, čo je neporovnateľne viac oproti slovenskému slovníku. Tu môžeme vidieť silu konečného automatu, ktorý dokázal uložiť český slovník na niekoľkonásobne menšom priestore (viď Tabuľka 6.2).

**Tabuľka 6.2:** Porovnanie slovenského a českého slovníka

| | **slovenský slovník** | **český slovník** |
|---|---|---|
| **textová forma** [kB] | 60 774 | 1 774 270 |
| **konečný automat** [kB] | 859 | 4 137 |

### 6.4.2 Časová náročnosť

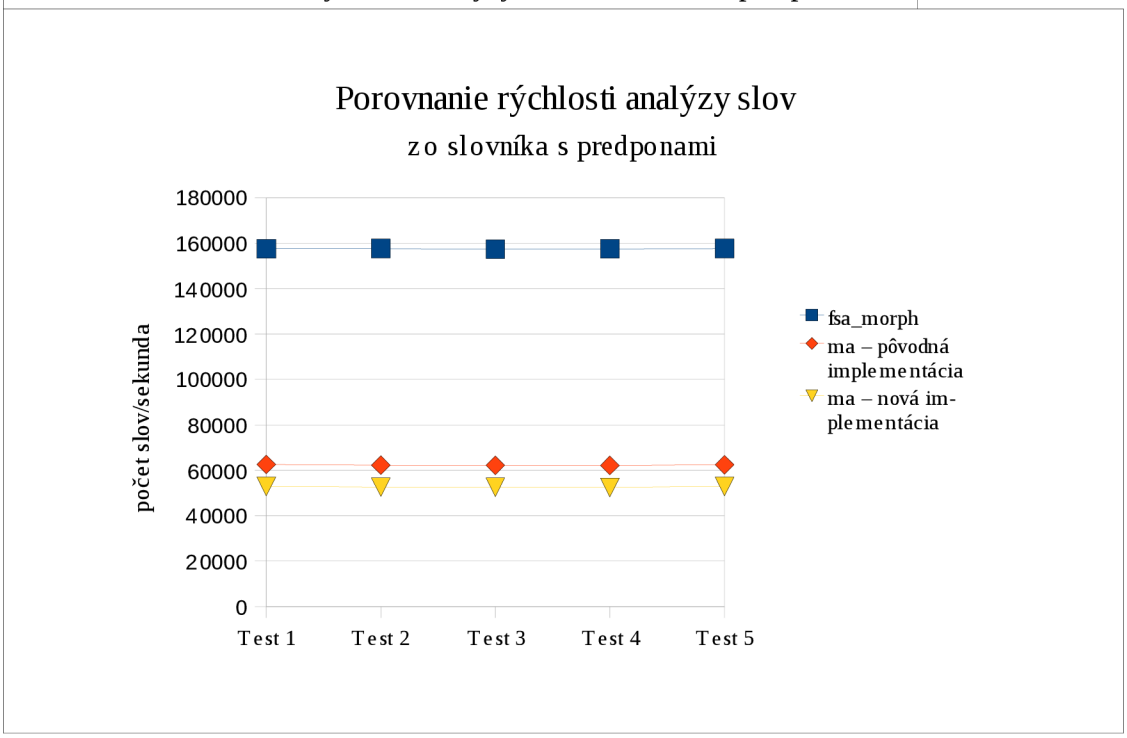
Zásahy do implementácie určite ovplyvnili aj beh aplikácie, preto je potrebné zistiť ako sa zmenil výkon a rýchlosť behu aplikácie.

Za testovacie dáta bola vybraná zbierka textov z internetu uložená v jednom súbore tak, že jeden riadok súboru obsahuje jediné slovo. Súbor mal veľkosť 29 826kB a obsahoval 3 900 733 slov. Pri každej funkcii bola vykonaná séria piatich testov, aby výsledok čo najlepšie odrážal skutočnosť. Pri testoch bol meraný čas, za ktorý stihne program slová z testovacieho súboru spracovať. Zaujímavejšie je pre nás však to, koľko slov stihne program spracovať za jednu sekundu, čo sa dá z nameraných časov jednoducho vypočítať (počet všetkých slov/čas).

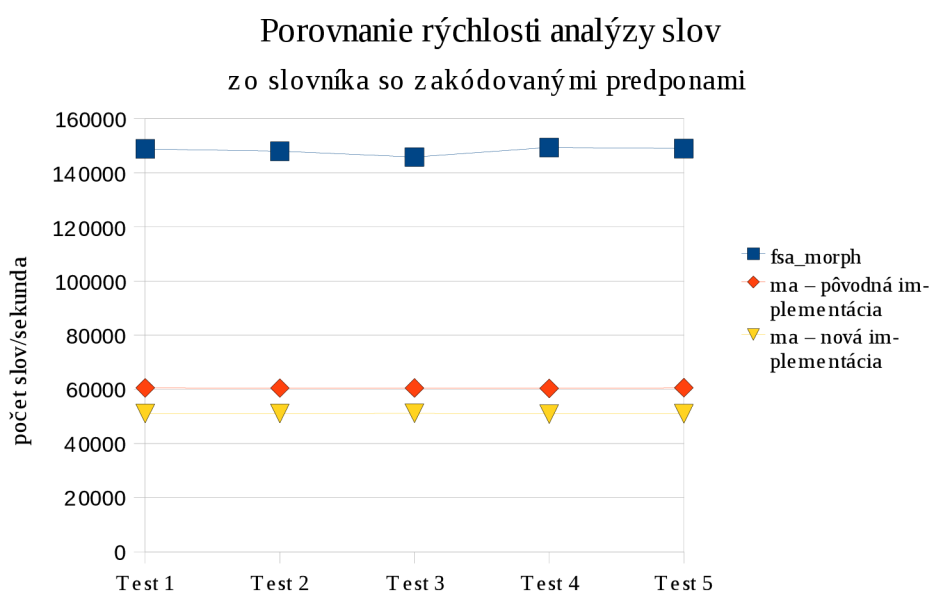
Boli vykonané nasledovné testy:

1. porovnanie rýchlostí analýzy slov rôznych implementácií (v podstate ich *lemmatizácie*), pričom boli použité dva slovníky, prvý so všetkými slovami, druhý so zakódovanými predložkami
2. porovnanie rýchlosti novej a pôvodnej verzie rozhrania pri generovaní všetkých tvarov s použitím slovníka bez gramatických kategórií a opakovania tvarov
3. porovnanie rýchlosti analýzy s dopĺňaním diakritiky pri generovaní všetkých tvarov a použitím s gramatickými kategóriami
4. porovnanie rýchlosti analýzy českého a slovenského jazyka

**Obrázok 6.1** Porovnanie rýchlosti analýzy slov zo slovníka s predponami



**Obrázok 6.2** Porovnanie rýchlosti analýzy slov zo slovníka s zakódovanými predponami



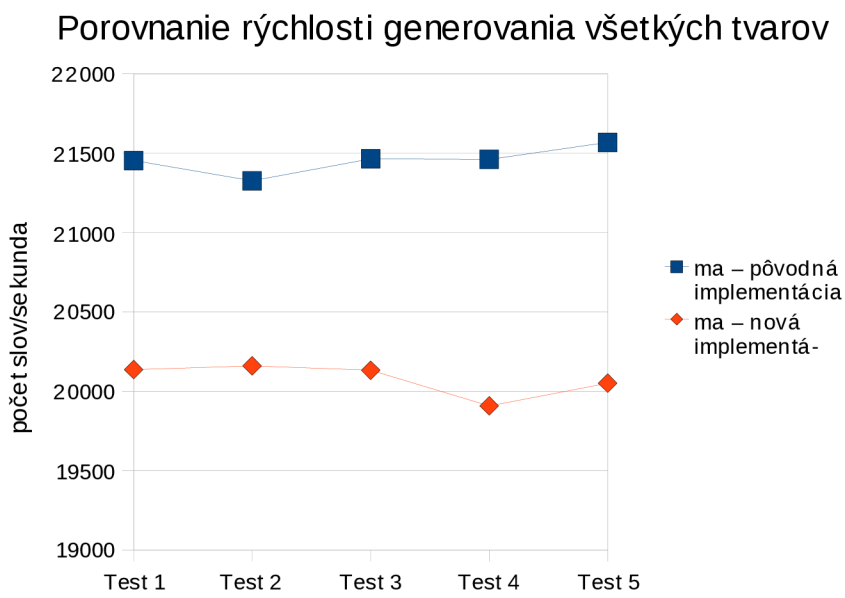

V grafoch na obrázkoch 6.1 a 6.2 vidno, že najrýchlejšia je implementácia z balíka *fsa*, *fsa_morph*, ktorá dokáže za sekundu spracovať viac než 150 000 slov. Spomalenie novej implementácie rozhrania oproti pôvodnej je minimálne, pričom používanie nového rozhrania je priateľnejšie. Ďalším sledovaným javom bol vplyv zakódovania predpôn do dát. Vtedy samotnej morfologickej analýze slova predchádza hľadanie predložiek, čo celý proces analýzy trochu spomalí. Ako je však vidieť pri pohľade na grafy a výslednú tabuľku 6.3, toto spomalenie nie je až také markantné, preto je výhodné ušetriť miesto na uloženie slovníkov a predložky do nich zakódovať.

Pre ďalšie porovnanie môžeme použiť dokument [3], v ktorom je popísaný nástroj *Morphonary*. Tento nástroj dokáže na základe iných princípov popísaných v kapitole 2 *lemmatizovať* 690 slov za sekundu, čo značne zaostáva za slovníkovým prístupom.

**Tabuľka 6.3:** Rýchlosť analýzy jednotlivých implementácií

| | fsa_morph | ma – pôvodná implementácia | ma – nová implementácia |
|---|---|---|---|
| predpony v slovníku [slov/s] | 157 577,42 | 62 322,6 | 52 656,76 |
| zakódované predpony [slov/s] | 148 139,49 | 60 494,43 | 51 007,59 |

**Obrázok 6.3** Porovnanie rýchlosti analýzy slov pri generovaní všetkých tvarov

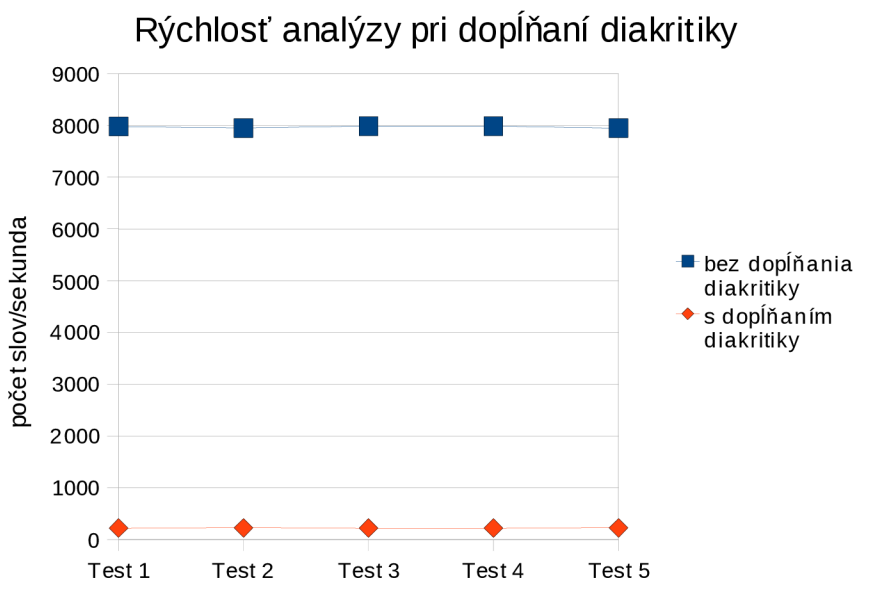



Nasledujúci test sledoval opačný proces ako je proces lemmatizácie, a tým je generovanie všetkých tvarov. Generovanie všetkých tvarov je o viac ako o polovicu pomalšie ako proces *lemmatizácie*, čo je vidno pri pohľade na grafy 6.2 a 6.3, prípadne tabuľky 6.3 a 6.4. Je to spôsobené aj tým, že rozhranie dokáže generovať tvary z rôznych nie len zo základného tvaru, ale aj z rôznych iných vygenerovaných tvarov. Z toho vyplýva, že pri použití slovníka, ktorý obsahuje generovanie slov od *lemmy* k tvaru, je nutné pred samotným generovaním tvarov získať tvar základný, teda zapojiť proces *lemmatizácie*. Nová implementácia opäť nezaostáva za pôvodnou a obe dokážu spracovať niečo nad 20 000 slov za sekundu.

**Tabuľka 6.4:** Rýchlosť generovania všetkých tvarov

| | ma – pôvodná implementácia | ma – nová implementácia |
|---|---|---|
| *ma -a* [slov/s] | 21 454,35 | 20 007,74 |

**Obrázok 6.4** Porovnanie rýchlosti analýzy pri dopĺňaní diakritiky

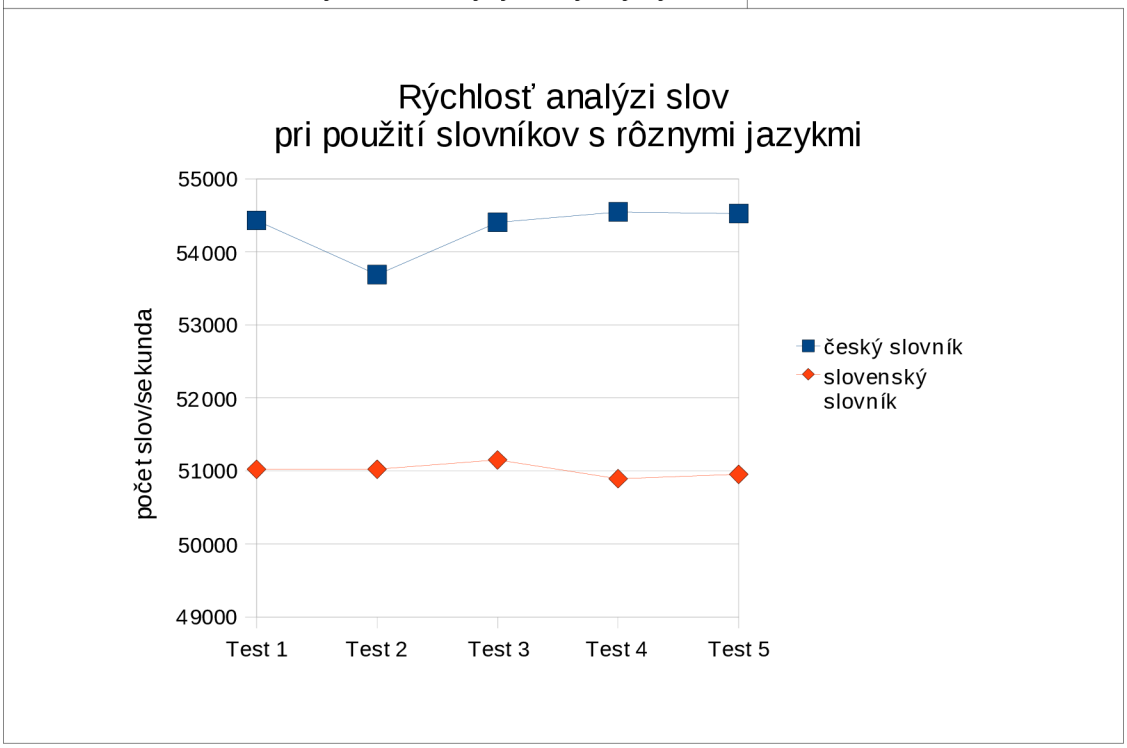



Doplňovanie diakritiky je veľmi komplikovaný proces, čo sa odrazilo aj na výkone celej aplikácie. Testovanie bolo vykonané v móde rozhrania, ktorý generuje všetky tvary aj s gramatickými kategóriami a umožňuje na základe nich radiť a filtrovať výstup aplikácie. Na testovanie dopĺňania diakritiky bol použitý menší súbor, ktorý obsahoval len 10 000 slov. Ako je vidno v grafe na obrázku 6.4 a v tabuľke 6.5, dopĺňanie diakritických znamienok nie je vhodné pri spracovávaní rozsiahlych textov, ale svoje využitie nájde skôr pri manuálnom zadávaní jednotlivých slov.

**Tabuľka 6.5:** Rýchlosť dopĺňania diakritiky

| | bez dopĺňania diakritiky | s dopĺňaním diakritiky |
|---|---|---|
| *ma -s* [slov/s] | 7 969,18 | 220,35 |

**Obrázok 6.5** Porovnanie rýchlosti analýzy rôznych jazykov




Cieľom tohto testu bolo zistiť závislosť rýchlosti spracovania dát od použitého jazyka. Výkon môže ovplyvniť najmä počet znakov abecedy v jednotlivých jazykoch. Keďže oba porovnávané jazyky majú takmer rovnako veľké abecedy, rýchlosť spracovania je tiež podobná. Pri testovaní českého jazyka bol použitý testovací súbor, ktorý obsahoval podobne ako slovenský testovací súbor 3 900 733 slov.

**Tabuľka 6.6:** Rýchlosť analýzy rôznych jazykov

| | český jazyk | slovenský jazyk |
|---|---|---|
| *lemmatizácia* [slov/s] | 54 317,71 | 51 007,59 |

# 7 Záver

Cieľom tejto práce bolo zoznámiť sa s existujúcimi prostriedkami na spracovanie českého jazyka a navrhnúť a realizovať morfologický analyzátor slovenčiny.

Dôležitou súčasťou morfologickej analýzy slovenského jazyka je proces *lemmatizácie*, čiže získavania základného tvaru. Tento proces je v našom, ale aj iných slovanských jazykoch komplikovaný oproti iným svetovým jazykom.

Pri návrhu morfologického analyzátora budeme vždy potrebovať menší alebo väčší slovník, ktorý je potrebné uložiť tak, aby zaberal, čo najmenší priestor. Vhodnou štruktúrou je n-nárny strom TRIE, ktorého podobnosť s konečnými automatmi sa dá využiť na jeho minimalizáciu, ktorou dosiahneme optimálne uloženie rozsiahleho slovníka jazyka.

V slovenskom slovníku je zatiaľ oproti českému zahrnutých veľmi málo slov. Ak by sa aj podarilo do neho zahrnúť celú slovnú zásobu jazyka, práca na dátach by neskočila, pretože slovná zásoba sa neustále obohacuje o nové výrazy.

Cieľom v pokračovaní práce na morfologickom analyzátore, je vytvorenie univerzálneho analyzátora, ktorý bude schopný spracovať viacero jazykov, ktorých voľba bude závisieť len od zadaného slovníka. Zaujímavé by bolo spojiť morfologický analyzátor s programom, ktorý by dokázal analyzovať texty z hľadiska vetného rozboru. Tak by sa dali presnejšie určiť gramatické kategórie tvarov, zložených z viacerých slov napríklad u slovies – *povedal som*, *povedal si*, *povedal by som*, ... Kombináciu týchto dvoch programov by bolo možné použiť ako súčasť programu na kontrolu pravopisu, čo by mohlo viesť k presnejšiemu a užšiemu výberu slov ponúkaných ako náhradu za chybné slovo na základe kontextu.

# Literatúra


[1] DVONČ, Ladislav, et al. *Morfológia slovenského jazyka*. 1. vyd. Bratislava : Vydavateľstvo Slovenskej akadémie vied, 1966. 896 s.

[2] ŠMERK, Pavel. Morphemic Analysis: A Dictionary Lookup Instead of Real Analysis. In Sojka, Petr - Horák, Aleš. *First Workshop on Recent Advances in Slavonic Natural Languages processing*, RASLAN 2007. Brno : Masaryk University, 2007. ISBN 978-80-210-4471-5, pp. 77-85. 2007, Karlova Studánka, Czech Republic.

[3] KRAJČÍ, Stanislav, et al. The tool Morphonary: Using of word lemmatization in processing of documents in Slovak. In Návrat, P. - Chudá, D. (Eds.). *Znalosti 2009*, pp. 199-130, ISBN 978-80-227-3015-0. STU Bratislava, Fakulta informatiky a informačných technológií, 2009.

[4] ČERNÝ, Stanislav. *Morfologický analyzátor pomocí konečných automatů*, bakalářská práce, Brno, FIT VUT v Brně, 2008.

[5] DACIUK, Jan. Experiments with Automata Compression. In Yu, S. - Păun, A. (Eds.): *CIAA 2000*, LNCS 2088, pp. 105–112, 2001. Springer-Verlag Berlin Heidelberg 2001.

[6] RIPKA, Ivor, IMRICHOVÁ, Mária, SKLADANÁ, Jana. *Príručka slovenského pravopisu pre školy a prax*. Bratislava : OTTOVO NAKLADATEĽSTVO, 2005. 672 s. ISBN 80-969159-1-6.

[7] BENKO, Ján. *Stredoškolské učivo slovenského jazyka a literatúry : Tvorenie slov* [online]. c2007 [cit. 2009-05-10]. Dostupný z WWW: <http://ucivosjl.webgarden.cz/>.

[8] SEDLÁČEK, Radek. *Morfologický analyzátor češtiny*. Master's thesis, FI MU v Brně, Brno, 1999.

[9] SEDLÁČEK, Radek. *Morphemic Analyser for Czech*. PhD thesis, FI MU in Brno, Brno 2004

# Zoznam príloh

# Príloha 1 – Nápoveda rozhrania *ma*

```
./ma [-C file] [-m] [-n] [-a] [-f [tag]|-F] [-r] [-d [tag]|-D] [-b [tag]|-B]
./ma [-C file] [-l|-L|-i|-I]
./ma [-h]

Inflectional morphology:
 -m       .. print out morphological categories (default)
 -n       .. print out numeric value
 -a       .. print out all words
 -s       .. print out all words (sortable according to sort key in configuration file)
 -f [tag] .. print out form according tag
 -F       .. print out all forms (same as -a but little bit slower)

Derivational morphology:
 -r       .. print out roots
 -d [tag] .. print out derivated forms according tag
 -D       .. print out all derivated forms
 -b [tag] .. print out base forms according tag
 -B       .. print out all base forms

Is included or lemma? (can't be combined with other switches)
 -i       .. print out word only if it is NOT included in dictionary
 -I       .. print out word only if it is included in dictionary
 -l       .. print out word only if it is NOT lemma
 -L       .. print out word only if it is lemma

Options:
 -C       .. config file (default: "dict.conf")
 -V       .. verbose on input (default: off)
 -c       .. switch to corpus mode (default: off)
 -h       .. print out this help

Version of libma: 0.3c
```

# Príloha 2 – Príklad konfiguračného súboru

```
# Language file
language = "../lang/sk.lang"

# Accent file
accent = "../lang/sk.acc"

# Add diacritics
# 0 - don't add diacritics
# 1 - add diacritic
accent_level = 0

# Level of base form
# 0 - without prefixes
# 1 - with prefixes
lemmatization = 0

# Letters case
# 0 - exactly same as word in dictionary
# 1 - first letter could be changed to upper case
# 2 - all of above and capitalized word will be accepted too
# 3 - case of letter doesn't matter
case = 3

dictionary = "../data/morph_pre_sk.fsa"

numbers = "../data/numbers.fsa"

base2form = "../data/all_words_sk.fsa"

base_annot2form = "../data/get_form_sk.fsa"

derivation = "../data/deriv.fsa"
```

```
sort_key = "NBQ"
filter_key = "k1c1nS"

# Filter
# 0 - filter IN  -> output only words with same categories as filter_key
# 1 - filter OUT -> output words, excluding the ones with the same categories as filter_key
filter_type = 0
```

Vysvetlenie skratiek vo vyhľadávacom kľúči (*sort_key*):

A: vid slovies

B: pád

C: gramatické kategórie ako reťazec

D: stupeň prídavných mien a prísloviek

E: tvar negovaný/nenegovaný

F: tvar

G: rod

M: slovesný spôsob

N: číslo

P: vzor

Q: osoba u slovies

T: terminál

W: štylistický príznak tvarov

X: zaradenie slovného druhu 1

Y: zaradenie slovného druhu 2

# Príloha 3 – Príklad súboru s akcentačnou tabuľkou

```
# lower case
a áä
c č
d ď
e é
i í
l ĺľ
n ň
o óô
r ŕ
s š
t ť
u ú
y ý
z ž
#
# upper case
A ÁÄ
C Č
D Ď
E É
I Í
L ĹĽ
N Ň
O ÓÔ
R Ŕ
S Š
T Ť
U Ú
Y Ý
Z Ž
```