Nalezení slovních kořenů v češtině

Petr Chmelař[1], David Hellebrand[1], Michal Hrušecký[2] a Vladimír Bartík[1]

[1]FIT, Vysoké učení technické v Brně,
Božetěchova 1/2, 612 66, Brno
[2]MFF, Univerzita Karlova v Praze
Ke Karlovu 3, 121 16, Praha 2
chmelarp@fit.vutbr.cz, david.hellebrand@seznam.cz, michal@hrusecky.net,
bartik@fit.vutbr.cz

Abstrakt. Cílem bylo vytvořit stematizační algoritmus pro český jazyk založený na gramatických pravidlech jako doplněk k metodám používajícím slovník pro vyhledávání a dolování českého textu. Článek obsahuje základy slovotvorby českého jazyka pro různé slovní druhy, popis problematiky stematizace a několika stematizačních a lematizačních algoritmů. Hlavním přínosem této práce je Snowball implementace stematizačního algoritmu českého jazyka na základě kompletní sady všech předpon a přípon, které se mohou v českém jazyce vyskytovat.

Klíčová slova: Lemmatizace, stematizace, Snowball, český jazyk, gramatika.

1 Úvod

Stematizace je proces určování základu slova (tzv. *stem*). Výsledný stem se nemusí nutně shodovat s lingvistickým kořenem slova ačkoliv v mnoha případech se shodovat budou. V českém jazyce, který má bohaté skloňování slov, se jedná o často užívanou metodu. Lematizace je proces určování základního tvaru slova (tzv. *lemma*) používaného například ve slovnících. Nalezení gramatického kořene slova nebo lemmatu v češtině je často algoritmicky složité. Navíc v reálných aplikacích je často cílem jen nalezení takové části slova, která bude stejná pro všechna příbuzná slova (stem). I proto je stematizace velmi užitečná.

Stematizační algoritmy mohou být využity zejména v oblastech získávání informací (*information retrieval*). Typickým příkladem jsou např. webové vyhledávače. U vyhledávačů může mít stematizace ovšem za určitých podmínek také negativní dopad, protože rozšiřuje množinu hledaných slov a tak se ve výsledcích vyhledávání mohou objevit i nežádoucí dokumenty.

Nabízí se také využití při lemmatizaci, kdy nám stemmer sjednotí všechna příbuzná slova a následně se už jen ze stemu vytvoří lemma (zde je ovšem třeba rozlišit různá slova mající stejný stem). Proto se pro lemmatizaci i stematizaci používá mnoho společných postupů. Navíc v některých jazycích s méně složitou morfologií mohou algoritmy níže popsané vést k nalezení kořenu nebo i lemmatu daného slova přímo.

Z historického hlediska se stemmery dělily buď na slovníkové, nebo algoritmické. V současnosti ale nejsou tyto dva přístupy tak přísně rozděleny. Algoritmický stemmer může obsahovat dlouhé seznamy výjimek, které můžeme chápat jako účinné slovníky. Stejně tak i slovníkový stemmer potřebuje umět odstraňovat alespoň některé přípony, aby vůbec bylo možné daná slova ve slovníku vyhledávat.

Článek je založen na diplomové práci Davida Hellebranda pod vedením Petra Chmelaře. Cílem této práce bylo implementovat v jazyce Snowball stemmer českého jazyka založený na gramatických pravidlech, který nemusí nutně umět najít přesný morfologický kořen slova, ale měl by všechny existující tvary slova převádět na stejný tvar (stem).

2 Stematizace

Stematizace (stemming) je postup, během kterého se slova převádějí na jejich základ – tzv. stem. Výsledný stem se nemusí nutně shodovat s morfologickým kořenem slova. Obvykle je dostačující, pokud jsou příbuzná slova převedena na stejný tvar, i když tento tvar není přesným lingvistickým kořenem slova.

Základní myšlenkou stematizace je podle [1] snaha o vylepšení procesu získávání informací tím, že k několika slovům majícím podobný význam určí jejich společný základ, ze kterého byly odvozeny.

Stematizaci není možné provádět s úplně všemi jazyky. Například s čínštinou si neporadí. Naproti tomu u jazyků patřících do Indo-Evropské rodiny jazyků jsou jednotlivá slova tvořena na základě různých gramatických pravidel, takže je možné na ně použít některý stematizační algoritmus.

Jako *stem* budeme označovat výsledek procesu stematizačních algoritmů. Naproti tomu *kořen slova* budeme chápat opravdu jako gramatický slovní kořen. Tedy jako základní tvar daného slova.

Pro rodinu Indo-Evropských jazyků je typické, že základem každého slova je kořen slova. Před ním bývají umístěny předpony (*prefixy*), za ním přípony (*sufixy*). Předpony i přípony mají souhrnný název *afixy*. V [1] je uvedeno, že přípony můžeme rozdělit do tří základních skupin: D-, I- a A-přípony.

A-přípona (*attached suffix*) je slovo připojené k jinému slovu. Tento typ přípon se často objevuje v italštině, španělštině a také v portugalštině (ačkoliv v portugalštině bývá od předcházejícího slova oddělen pomlčkou, takže je snadné je odstranit).

Např. v italštině se osobní zájmeno připojuje ke slovesu:

- mandargli = mandare + gli = poslat + jemu

I-přípona (*inflectional suffix*) vychází ze základních pravidel gramatiky daného jazyka. Je možné ji použít na všechna slova určitého slovního druhu (i když může existovat malé množství výjimek). Např. v angličtině se minulý čas sloves tvoří přidáním přípony „ed“. V některých případech je nutné navíc upravit tvar kořene slova.

- fit + ed = fitted (zdvojené „t“)

– love + ed = loved (vypuštění koncového „e“ ve slově „love“)

V případě I-přípon je také poměrně běžné, že mohou mít více funkcí. Např. v angličtině může přípona „s“ znamenat:

– Sloveso ve třetí osobě jednotného čísla (runs, sings)
– Množné číslo podstatného jména (dogs, cats)
– Podstatné jméno označující vlastnictví (boy's, girl's)

Ale jinak je možné toto pravidlo použít na všechna slovesa současné angličtiny, kromě zhruba 150 nepravidelných sloves (např. become – became, begin – began, atd.).

D-přípona (*derivational suffix*) umožňuje z jednoho slova vytvořit slovo jiné, častokrát jiného slovního druhu nebo jiného významu. Může-li být určitá D-přípona přidána k nějakému slovu, nelze zjistit na základě pravidel gramatiky daného jazyka, ale pouze ve spolupráci se slovníkem.

Vzhledem k tomu, že ve slově za sebou přípony zpravidla následují v pořadí D, I a A, budou zpravidla odstraňovány zprava v pořadí A, I a D. Obvykle se snažíme o odstranění všech A- a I-přípon a alespoň některých D-přípon.

2.1 Stematizační a lemmatizační algoritmy

Existuje několik základních typů algoritmů použitelných jako základ pro lemmatizaci či stematizaci. Navzájem se od sebe liší jednak výkonností (rychlost nalezení výsledků), přesností nalezených výsledků a také schopností, jak se vypořádat s překážkami číhajícími v procesu lemmatizace/stematizace daného jazyka.

Brute Force algoritmy [2] nevyužívají pro svou práci žádných vlastností jazyka. Využívají pouze tabulku, ve které jsou uvedeny dvojice kořen slova – vyskloňovaný tvar slova. Zjištění kořene slova probíhá tak, že se dané slovo hledá v této tabulce. Pokud je nalezeno, vrátí algoritmus příslušný kořen slova nalezený v tabulce. Pokrytí celého jazyka je tedy zásadním problémem pro vylepšování Brute Force algoritmů.

Suffix stripping algoritmy [2], na rozdíl od Brute Force algoritmů, nepotřebují tabulku s výčtem všech slov jazyka a k nim příslušných kořenů slov. Na místo toho jim stačí relativně malý seznam pravidel, na jejichž základě z daného slova odstraní případné přípony vyskytující se v daném jazyce. Vývoj a vylepšování těchto algoritmů je v porovnání s Brute Force algoritmy mnohem jednodušší, ovšem za předpokladu, že vývojář má v požadovaném jazyce dostatečné znalosti v oblasti jazykovědy a morfologie. Problémem je u těchto algoritmů zpracování různých výjimek v jazyce. Na ně totiž obecná pravidla tvorby slov v daném jazyce neplatí a algoritmus si tedy nemá s těmito výjimkami jak poradit. Nasazení suffix stripping algoritmů je tedy omezeno pouze na slovní druhy (podstatné jméno, přídavné jméno, sloveso atd.), u kterých jsou v daném jazyce známy možné přípony a neexistuje pro ně mnoho výjimek.

U jednotlivých suffix stripping algoritmů se mohou poskytované výsledky z několika důvodů lišit:

– Prvním důvodem je podmínka, zda výsledný stem musí být platným slovem v daném jazyce. Některé přístupy toto nevyžadují. Jiné si zase existenci výsledného stemu kontrolují v databázi všech platných kořenů slov daného jazyka. Pokud se v ní výsledné slovo nevyskytuje, provede algoritmus alternativní kroky (např. zkusí odstranit jinou příponu). Pro takovéto případy může mít algoritmus jednotlivým pravidlům na odstraňování přípon přiřazeny různé priority (buď zadané člověkem, nebo získané stochasticky).
– Další možností může být, že se algoritmus rozhodne neaplikovat takové pravidlo, jehož výsledkem není platné slovo daného jazyka, pokud má k dispozici nějaké jiné pravidlo dávající platný výsledek.

Další algoritmy (*Stematisation Algorithms*) [2] se snaží o komplexnější přístup k nalezení kořene slova. Nejdříve se pokouší zjistit, o jaký slovní druh se u hledaného slova jedná, a na základě toho poté aplikují pro každý slovní druh rozdílné normalizační metody a pravidla pro nalezení stemu.

Stochastické algoritmy (*Stochastic Algorithms*) [2] využívají pro nalezení kořene slova pravděpodobnost. Nejdříve u nich probíhá tzv. trénovací fáze, během níž si ze vztahů mezi kořenem slova a z něj odvozených slov vytváří pravděpodobnostní model. Tento model zpravidla představuje množinu pravidel podobnou jako u Suffix stripping algoritmů. Hledání kořene slova potom probíhá tak, že natrénovanému modelu je předloženo dané slovo a model na základě své naučené množiny pravidel nalezne nejpravděpodobnější kořen daného slova.

V [1] je uvedeno, že z historického hlediska se stematizátory dělily buď na slovníkové, nebo algoritmické. V současnosti ale nejsou tyto dva přístupy tak přísně rozděleny. Algoritmický stematizátor může obsahovat dlouhé seznamy výjimek, které můžeme chápat jako účinné mini slovníky. Stejně tak i slovníkový stematizátor potřebuje umět odstraňovat alespoň I-přípony, aby vůbec bylo možné daná slova ve slovníku vyhledávat.

Hybridní algoritmy [2] kombinují několik výše uvedených přístupů. Podobně jako Brute Force algoritmy mohou nejdříve zkontrolovat, jestli se dané slovo nenachází v jejich databázi. Ovšem na rozdíl od Brute Force si v databázi neuchovávají všechna slova daného jazyka, ale jen často se vyskytující nepravidelné tvary slov. Tabulka tedy zůstává malá a je možné v ní rychle vyhledávat. Pokud se hledané slovo v tabulce nenachází, přichází na řadu další algoritmy, např. Suffix stripping.

2.2 Chybové metriky

V [2] je uvedeno rozdělení chybových metrik na *under-stemming* a *over-stemming*.

over-stemming znamená, že dvě různá slova dostanou algoritmem přiřazený stejný stem, i když by tomu tak být nemělo – *false positive*. Jinými slovy chyba vzniklá odebráním příliš velké části přípony.

under-stemming znamená, že dvěma různým slovům by měl algoritmus přiřadit stejný stem, ale není tomu tak – *false negative*. Neboli chyba vzniklá odebráním příliš malé části přípony.

Stematizační algoritmy by se měly snažit minimalizovat obě chyby. Avšak omezení jedné chyby má většinou za následek nárůst druhé. Dále v článku míru chybovosti určujeme použitím slovníku [3], i když ani kvalitní slovník nám nezaručí 100% úspěšnost.

3 Morfologie češtiny

Čeština je západoslovanský jazyk, stejně jako např. slovenština a polština. Patří do rodiny indoevropských jazyků, konkrétně mezi slovanské jazyky.

V [4] se uvádí, že morfologie (tvarosloví) je lingvistická disciplína tvořící součást klasické gramatiky. Studuje všechny typy slovních druhů z hlediska jejich formy a funkce.

Podle [5] se formou slova roumějí všechny jeho podoby – tvary, jichž se užívá ve větách. Tyto formy se skládají z tvarotvorného základu, který je společný pro všechny tvary slova, a z morfu.

Morfem rozumíme útvar nesoucí gramatický význam, a to především koncovku (*žen-y, píš-u*) včetně koncovky nulové. Morfy mohou být součástí slova, nebo jsou samostatné (volné): *bych, bys* atd. Tvary slov se stejnými soubory morfů tvoří typ (vzor) nebo podtyp.

3.1 Slovní druhy

V češtině se rozlišuje 10 slovních druhů: podstatná jména (substantiva), přídavná jména (adjektiva), zájmena (pronomina), číslovky (numeralia), slovesa (verba), příslovce (adverbia), předložky (prepozice), spojky (konjunkce), citoslovce (interjekce) a částice (partikule).

Podle [5] lze slovní druhy dělit na ohebné a neohebné, v rámci ohebných pak na skloňované (substantiva, adjektiva, zájmena a číslovky) a časované (slovesa). Neohebné slovní druhy dále nediskutujeme. Toto dělení však není absolutní a mohou se vyskytovat výjimky.

Substantiva (podstatná jména) V [4] se uvádí, že substantiva jsou základním slovním druhem označujícím jevy skutečnosti jako samostatná fakta. Substantiva zahrnují nejen jména osob, zvířat, věcí, jevů, ale i vlastnosti a děje pojímané jako nezávislé entity (*krása, běh*). Substantivum je ohebný slovní druh, který prostřednictvím systému tvarů vyjadřuje mluvnické významy pádu, čísla a rodu. Kromě několika málo výjimek (např. slovo „nůžky“) všechna podstatná jména existují v jednotném i v množném čísle.

Podle [5] je gramatický rod závazný pro každé jméno a zároveň je klasifikačním prostředkem. O přirozený rod se opírá dělení na maskulina, femina a neutra (mužský, ženský a střední rod). Zakončení substantiv na konsonant (souhlásku)

je obvykle znakem maskulin (kromě typů *předseda, soudce, hajný, krejčí*). Zakončení na -a, -e/-ě je typické pro feminina kromě typů *píseň, kost, pokojská, vedoucí*. Zakončení na -o, -e/-ě je typické pro neutra kromě typů *stavení, jízdné, telecí*.

Čeština má sedm pádů (nominativ, genitiv, dativ, akuzativ, vokativ, lokál, instrumentál – tradičně se označují jako 1. – 7. pád). Jejich přehled je v [5]. Jednotlivé vzory jsou tvořeny koncovkami sedmi pádů, a to v jednotném i množném čísle. Žádný vzor však neobsahuje čtrnáct různých koncovek, existuje tu jistá homonymie. Nejvíce různých koncovek má vzor *žena*. U mužského rodu se tyto vzory dělí dále na životné a neživotné. Běžně se uvádí 31 vzorů (z toho 14 základních), nicméně vzhledem k výjimkám, jich při počítačovém zpracování může být až sto.

Verba (slovesa) Definice v [4] uvádí, že sloveso je ohebný plnovýznamový slovní druh. Označuje dynamické (v čase probíhající) příznaky substancí. Sloveso může vyjádřit děj, který v okamžiku promluvy proběhl (děj minulý), nebo děj probíhající (přítomný) anebo děj, který proběhne (budoucí) [5]. Tvary slovesné vyjadřující osobu, čas a způsob jsou v [5] nazývány *určitými tvary*. Čeština má tři gramatické osoby, kmeny a časy a osm nepravidelných sloves.

Adjektiva (přídavná jména) Podle [4] jsou adjektiva plnovýznamové slovní druhy označující vlastnosti substancí (tj. osob, zvířat a neživých jevů konkrétních i abstraktních). Rod, číslo a pád adjektiv je dán shodou se substantivem, ke kterému se vztahují. Z morfologického hlediska (tj. pokud jde o druhy skloňování) se rozlišuje trojí typ adjektivní flexe: jmenná, složená a smíšená (vzory *otcův, matčin*). Složená se pak dále dělí na tvrdou (vzor *mladý*) a měkkou (vzor *jarní*).

Pronomina (zájmena) Zájmena představují plnovýznamový ohebný slovní druh tvořený v podstatě uzavřenou množinou výrazů. Zájmena různou měrou realizují schopnost jazyka pojmenovávat substance a vlastnosti substitučně [4].

Další slovní druhy – číslovky nepevné ohraničení, příslovce, spojky, částice, citoslovce jsou dle [5] neohebné, proto je dále neuvažujeme. Čeština dále obsahuje dle slovníku [3] také 500 nepravidelných slov.

3.2 Slovotvorba

Slovotvorba je gramatická disciplína [4], zabývající se jednak procesem tvoření slov na základě slov již existujících z hlediska způsobů a postupů, jednak formou, významem a fungováním výsledků těchto procesů. Při vytváření slov mohou mít jejich složky povahu morfémů, a pak se jedná o derivaci (odvozování), nebo se může slovní základ kombinovat s jiným a jde o kompozici (skládání slov).

4 Návrh a implementace

Snowball [1] je malý jazyk pro práci s řetězci. Jeho jméno bylo zvoleno jako pocta jazyku SNOBOL. S tímto jazykem má také společnou základní myšlenku, a to že určité vzory řetězců vysílají signály, které jsou využívány k řízení běhu programu. Snowball je jazyk sloužící ke snadnému a přesnému popisu stematizačních a lematizačních algoritmů. Kompilátor jazyka Snowball překládá skripty na programy v jazyce Java nebo ANSI C.

V současné době existují stemmery v jazyce Snowball pro 16 různých světových jazyků [1], ale čeština mezi nimi není. Stemmer nemusí generovat přesný morfologický kořen slova, ale měl by všechny možné tvary jednoho slova převádět na stejný tvar (stem). Pro češtinu existují pouze dva brute force stemmery pracující se slovníkem, ve kterém mají ke každému slovu přiřazen jeho základní tvar a dále dva založené na pravidlech [6], nicméně tyto nepokrývají všechny zákoutí českého jazyka a dosahují úspěšnosti pouze 40%.

Vlastní implementace je odvozena od Snowball stemmeru pro angličtinu a němčinu [1]. Náš stemmer očekává na vstupu slova skládající se pouze z malých písmen v kódování UTF-8. Výstup je převáděn na malá písmena bez diakritiky.

4.1 Popis algoritmu

Na začátku programu se volá procedura `exception` obsahující různé výjimky a nepravidelně skloňovaná česká slova a jejich stemy. Pokud se zpracovávané slovo nachází v tomto seznamu, vrací program rovnou příslušný stem a nepokouší se aplikovat na slovo žádná pravidla.

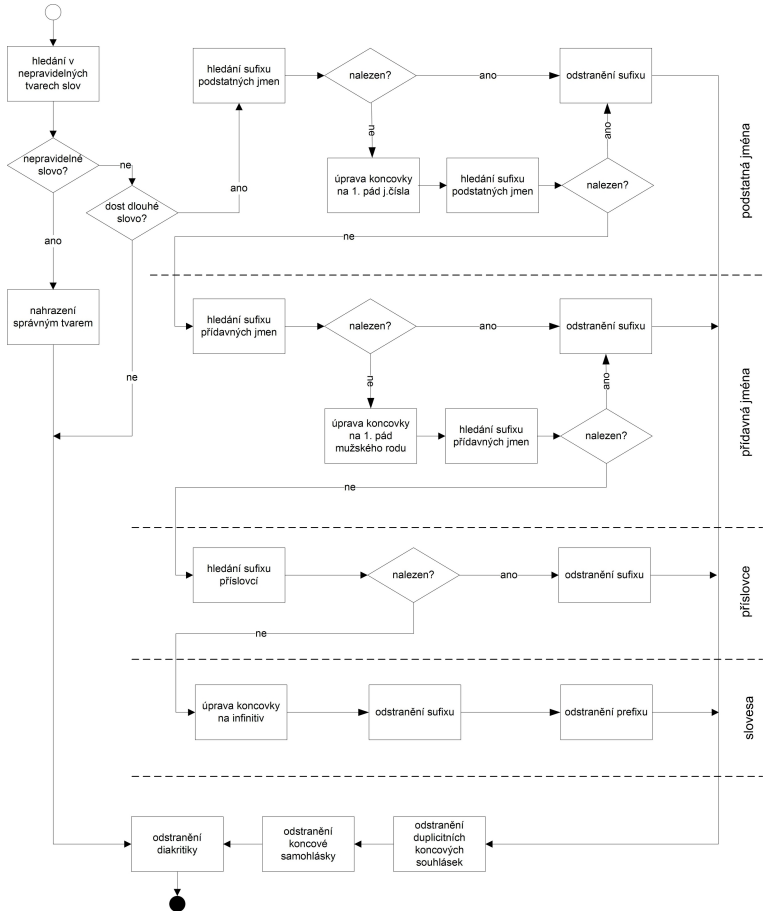
Když není slovo na vstupu zachyceno jako výjimka, pokračuje běh programu voláním procedury `mark_regions`. Ta má za úkol vybrat ze slova tzv. úsek R1. Jedná se o úsek začínající za první souhláskou, které předchází samohláska, a pokračující až do konce řetězce.

```
t  r  a  k  t  o  r  i  s  t  a
         |<------------------->|   R1
```

V uvedeném příkladu je písmeno *k* první souhláskou, před kterou stojí samohláska. Úsekem R1 bude tedy řetězec *'torista'*. Stemmer provádí veškeré operace se slovy jen nad jejich R1 částmi. Pokud tedy slovo takový úsek vůbec neobsahuje, je ponecháno beze změn a odesláno na výstup. Účelem tohoto opatření je eliminovat zpracování jednoslabičných slov. Jestliže nemá slovo více slabik, pak pravděpodobně nemohlo vzniknout přidáním prefixu nebo sufixu a je tedy zbytečné pokoušet se na něj aplikovat nějaká gramatická pravidla.

Program dále pokračuje voláním procedur, které na základě gramatických pravidel češtiny vytvoří ze vstupního slova jeho stem pro různé slovní druhy, jak je uvedeno dále a schematicky znázorněno na schématu v obrázku 1.

Na závěr stemmer ze slova odstraňuje případné koncové duplicitní souhlásky, případné koncové samohlásky a z výsledného stemu odstraní diakritiku.



Obr. 1. Schéma činnosti stemmeru češtiny.

Podstatná jména jsou zpravidla tvořena sufixací a dají se rozdělit do mnoha skupin, přičemž v každé skupině jsou slova tvořena přidáním různých sufixů. Např. skupina „jmen konatelských“ obsahuje sufixy *-ář, -íř, -ník, -ista, -ák, -ec*. Do skupiny „jmen obyvatelských“ patří sufixy *-an, -ec, -ák* atd. Celkem je ve stemmeru použito 267 různých sufixů. V dalším kroku se odstaní koncovky vznikající při skloňování podstatných jmen. Koncovky vycházejí ze 48 vzorů podstatných jmen a jejich podtypů, celkem 155 různých koncovek.

Podstatná jména sice nejsou tvořena prefixací v takové míře jako slovesa, ale i přesto jich je v [5] uvedeno zhruba 80. Odstraňování prefixů ale vedlo pouze ke zhoršení výsledků, algoritmus nebyl schopný rozlišit, kdy se skutečně jedná o prefix a kdy ne. Např. *pra-* ve slově *Praha, prapor*.

V dalším kroku jsme se pokusili pro vylepšení stemmeru využít český slovník pro Ispell [3]. Autoři sestavili databázi českých slov a ke každému přidali určité příznaky, které určují, jak se dané slovo v gramatice chová (např. generuje předponu ne-, generuje tvary podle vzoru stroj, generuje tvar s koncovkou -ě/-e pro slova vzoru město atd.). Nicméně sami autoři poznamenávají, že úspěšnost tohoto skriptu je nízká (kolem 40 %). Na základě tohoto slovníku je dále možné vygenerovat až 1 300 spojení suffixů a koncovek, nicméně testování nedopadlo úspěšně.

Předchozí způsob sice k úspěšnému konci nevedl, ale přesto zůstává převod slov do základního tvaru základním požadavkem pro správnou funkčnost stemmeru. Proto jsme na základě analýzy koncovek sestavili tabulku mapující koncovky v jednotlivých pádech na jejich tvar v první osobě jednotného čísla. Výsledkem bylo, že 60 % ze všech tvarů koncovek mělo jednoznačně přidělený tvar pro první osobu jednotného čísla. Např. každé slovo zakončené koncovkou *-bou* má v základním tvaru koncovku *-ba* (*oso/bou* → *oso/ba, klen/bou* → *klen/ba* atd.).

U zbývajících 40 % koncovek není možné jednoznačně rozhodnout, jaký základní tvar zvolit. Jednou z koncovek, u kterých toto nelze provést, je např. *-ovi* (starost/ovi → starost/a, ale pán/ovi → pán/0). Pokud neznáme kontext zpracovávaného slova, neexistuje žádný způsob založený na gramatických pravidlech, na základě kterého by se dalo rozhodnout o správné koncovce základního tvaru. Proto v implementaci stemmeru se tyto koncovky mažou.

Pokud se hned při prvním vykonání procedury odstraňující sufixy podaří některý sufix nalézt a odstranit, nebude dále volána procedura převádějící slovo do základního tvaru. Ze zbývající části slova se případně odstraní koncová samohláska (viz obrázek 1).

Výsledky testování algoritmu na sadě podstatných jmen:

```
správně:    1433        (61.9 %)
špatně:     881 z 2314 (38.1 %)
            under-stemming:  663
            over-stemming:   171
```

Přídavná jména jsou stejně jako jména podstatná tvořena hlavně sufixací, rozhodl jsem se vyzkoušet pro jejich stemování obdobný princip jako pro podstatná jména.

Implementace pracuje se 101 různými sufixy přídavných jmen. Na rozdíl od situace u podstatných jmen zde dochází k mnohem častějším alternacím. Převládají alternace *h* → *ž, ch* → *š, k* → *č*.

Koncovky přídavných jmen byly získány zanalyzováním 14 vzorů a jejich podtypů, celkem 126 různých koncovek. Na rozdíl od situace u podstatných jmen se zde podařilo u většiny slov podle jejich koncovky určit, jakou podobu má koncovka pro základní tvar (1. pád mužského rodu).

Výsledky testování algoritmu na sadě přídavných jmen:

```
správně:    595       (82.5 %)
```

```
špatně:     126 z 721 (17.5 %)
            under-stemming:   89
            over-stemming:    35
```

Slovesa se liší od postupů platných pro podstatná a přídavná jména. Při sufixaci se u víceslabičných sloves vychází z podoby jejich infinitivního kmene, proto se pokouším slovesa převádět do infinitivu. Mnohem větší roli než u ostatních slovních druhů hraje při vytváření sloves prefixace. Důležitým poznatkem je, že hláskové alternace jsou velmi vzácné.

Stemmer zpracovává celkem 15 sufixů a 29 prefixů. Při převodu slovesa na infinitiv se pracuje s 345 koncovkami, které byly získány z 18 vzorů. U většiny koncovek se podařilo jednoznačně určit, jak podle nich převést sloveso do infinitivu. Čeština však obsahuje i 8 nepravidelných sloves, proto je přímo ve zdrojovém kódu všem tvarům těchto sloves přiřazen správný infinitivní tvar.

Opět byl vyzkoušel stejný princip algoritmu, jako u podstatných. U sloves ale nedosahoval tak dobrých výsledků. První volání procedury hledající sufixy velmi často odstranilo nesprávný úsek slova. Proto se nejdříve na základě koncovky převádí sloveso do základního tvaru a až poté se odstraňuje sufix. Dále se odstraňuje případný prefix. Protože na konci výsledného stemu často dochází ke zdvojení některých souhlásek, jsou odstraněny.

Stemmer na sadě sloves dosahuje těchto výsledků:

```
správně:    347       (70.5 %)
špatně:     145 z 492 (29.5 %)
            under-stemming:   35
            over-stemming:    76
```

Příslovce nemají taková složitá pravidla jako předchozí slovní druhy, proto stemmer odstraňuje pouze 6 sufixů.

5 Experimentální výsledky

Protože stemmer nemá vstupní slovo zařazeno do kontextu, nezná o něm tedy žádné podrobnější informace (např. o jaký slovní druh se jedná, v jakém je čísle atd.) a nemůže tak rozhodnout, který stemmovací algoritmus použít. Proto jsme pořadí zpracování určili na základě četnosti výskytu jednotlivých slovních druhů v českých textech. Jedinou výjimku tvoří slovesa, jejichž stemmer se jeví jako agresivní, zpracovává i jiné slovní druhy, proto byl zařazen až na konec.

Výsledná podoba stemmeru byla testována na kombinaci všech předchozích testovacích dat. Vznikl tak soubor podstatných jmen, přídavných jmen a sloves o celkové velikosti 3 530 slov. Podle očekávání se správnost výsledku snížila, protože u některých slov docházelo ke zpracování špatnou části stemmeru (např. s přídavným jménem mohlo být zacházeno jako se slovesem apod.). Těmto situacím se však dá bez doplňujících informací jen těžko zabránit.

Výsledky činnosti stemmeru na uměle vytvořené sadě dat jsou následující:

```
správně:    2065        (58.5 %)
špatně:     1462 z 3527 (41.5 %)
            under-stemming:  746
            over-stemming:   588
```

Dále jsme provedli testování na datech vytvořených z novinových článků:

```
správně:    276       (48.2 %)
špatně:     297 z 573 (51.8 %)
            under-stemming:  108
            over-stemming:   155
```

V testovací sadě všech slovních druhů pohromadě existovalo každé slovo průměrně v 9,8 variantách. Zhruba 50 % slov se převedlo na maximálně dva různé stemy. Statistiku zhoršují především výsledky stemmování sloves a fakt, že některá slova byla stemmována podle pravidel pro jiný slovní druh.

6 Závěr

Během vývoje algoritmu nebyl znám jiný volně dostupný stemmer pracující na stejném principu. Pro český jazyk existují slovníkové stemmery založené zpravidla na datech z českého slovníku pro Ispell a dále statistický stemmer nečeských autorů [6] založený na vzdálenostních funkcích, který dosahuje úspěšnosti okolo 40%.

Testování nově vyvinutého stemmeru na reálných datech získaných z novinových článků ukázalo, že zhruba v 50 % případů nalezne správný gramatický kořen slova. U uměle sestavených testovacích dat snažících se pokrýt co možná nejširší část gramatických pravidel nalezne správný kořen slova přibližně v 60 % případů.

Dalším kritériem pro testování byl počet tvarů, na které stemmer převede všechny existující varianty daného slova. V tomto případě probíhalo testování pouze na uměle vygenerovaném seznamu slov. Téměř 80 % slov, která se vyskytovala průměrně v 9,8 různých variantách, se podařilo převést na maximálně 3 různé stemy. Výsledky na reálných datech by mohly být úspěšnější, protože testovací data záměrně obsahovala množství výjimek a málo používaných tvarů slov.

Finální implementace algoritmu[1] nyní pracuje jako pět za sebou následujících samostatných částí (zpracování nepravidelných slov, moduly pro podstatná jména, přídavná jména, příslovce a slovesa). Pořadí volání jednotlivých modulů je odvozeno na základě četnosti výskytu jednotlivých slovních druhů v češtině. Přesto v některých případech může dojít k tomu, že slovo bude zpracováno modulem pro jiný slovní druh, než by mělo být. Dalším problémem jsou některé hláskové alternace. Např. u sufixů začínajících písmenem „i“ se může koncové „ž“ ve slově měnit buď na „g“, nebo na „h“.

[1] Volně dostupný z http://www.fit.vutbr.cz/research/view_product.php?id=133

Řešením by mohlo být spojení všech modulů do jednoho nebo zakomponování některých pravidel z OpenOffice slovníku pro kontrolu pravopisu. Této skutečnosti by mohlo být využito pro ošetření některých chyb produkovaných stávajícím algoritmem. Jako nejvhodnější stemmer pro češtinu se jeví hybridní stemmer, který by měl k dispozici databázi s českou slovní zásobou, nicméně toto přesahuje možnosti jazyka Snowball.

Poděkování

Tato práce byla částečně podpořena grantem FIT VUT v Brně FIT-10-S-2, grantem GAČR P202/10/1333 a výzkumným záměrem MSM0021630528.

Reference

1. Porter M. F. Snowball: A language for stemming algorithms. http://snowball.tartarus.org/texts/introduction.html.
2. Přispěvatelé Wikipedie. http://en.wikipedia.org/wiki/Stemming.
3. Kolář P. Český slovník pro Ispell. ftp://ftp.tul.cz/pub/unix/ispell/.
4. Karlík P., Nekula M. a Pleskalová J. *Encyklopedický slovník češtiny.* Lidové noviny, 2002, ISBN 80-7106484-X.
5. Kolektiv autorů. *Příruční mluvnice češtiny.* Lidové noviny, 1995, ISBN 80-7106-134-4.
6. Dolamic L. a Savoy J. *Indexing and stemming approaches for the Czech language.* Information Processing & Management 45, č. 6, 2009.

Annotation:

Czech Stemming Algorithm

The goal was to create an algorithm for stemming Czech language based on grammatical rules, in addition to methods using vocabulary for retrieval and mining of Czech texts. The article includes the basics of Czech word formation for different word classes, description of problems and several stemming and lemmatization algorithms. The main contribution of this work is the implementation of the Snowball stemming algorithm for the Czech language based on complete sets of all prefixes and suffixes, which may occur in Czech words.