Univerzita Hradec Králové

Fakulta informatiky a managementu

Katedra informatiky a kvantitativních metod

Lemmatizace češtiny

Bakalářská práce

Autor: Dominik Bydžovský

Studijní obor: Informační management

Vedoucí práce: Mgr. Jiří Haviger, Ph.D.

Prohlášení

Prohlašuji, že jsem bakalářskou práci zpracoval samostatně a s použitím uvedené literatury.

V Hradci Králové dne 18.4.2017 Dominik Bydžovský

Poděkování

Děkuji vedoucímu bakalářské práce Mgr. Jiřímu Havigerovi, Ph.D. za metodické vedení, za podporu při psaní této bakalářské práce, za jeho cenné rady a trefné poznámky.

Anotace

Bakalářská práce přibližuje téma strojového zpracování češtiny a její specifika. Zodpovězenou otázkou je automatizování procesu lemmatizace, tj. převádění slov do jejich základního tvaru na základě algoritmů, a její transparentní využití v praxi.

Cílem práce je souhrnné představení softwarů, které lemmatizaci provádějí, jejich použití a selekce na základě definovaných kritérií pro nekomerční akademické účely. Pro vytvoření pořadí mezi konkurenty byl použit manažerský program pro podporu rozhodování Expert Choice.

Důraz byl kladen na praktickou funkčnost a použitelnost softwarů, které byly ozkoušeny na českých volných dílech rozdílné délky a na platformách Linux nebo Windows.

Ve výsledcích práce čtenář najde porovnání zkoumaných nástrojů, tabulku rychlostí a finální pořadí nástrojů, které odráží jejich kvalitu a použitelnost.

Klíčová slova

Lemmatizace, tokenizace, lemma, tagset, homonymie

Annotation

Title: Lemmatization of Czech

This Bachelor's thesis deals with the machine processing of the Czech language and other related specifics. The paper answers the question of automating the process of lemmatization, i.e. converting words into their basic forms based on algorithms, and its transparent use in practice.

The aim of this work is to provide a summary of all the software tools that perform lemmatization, their use and selection based on the defined criteria for non-commercial, academic purposes. Expert Choice, a management tool that supports decision making, was used to create a sequence among the competitors.

The emphasis was put on the practical functionality and usability of the software tools, which were tested on Czech public domain books of different lengths and on the Linux and Windows platforms.

The results provide the reader with the comparison of the examined tools, speed ranking table and the final ranking of the tools, which reflects their quality and usability.

Key words

Lemmatization, tokenization, lemma, tagset, homonymity

OBSAH

1. Úvod

Bakalářská práce se zabývá tématem strojového zpracování českého jazyka a **zasvěcuje** čtenáře do problematiky převádění slovních tvarů do jejich základního tvaru, tj. lemmatizace, a její použití v praxi. Zvláštní pozornost je věnovaná konceptům pro strojovou automatizaci zpracování textů, konkrétně metodě Ripple Down Rules, která je přenositelná mezi různými světovými jazyky.

Cílem práce je, kromě **obohacení** čtenáře o znalosti tématiky NLP (Natural Language Processing), **popsat**, **použít** vybrané lingvistické nástroje a na základě definovaných kritérií a s pomocí systému pro podporu rozhodování Expert Choice **vybrat** nejvhodnější nástroj.

Vytipované lingvistické nástroje jsou popsány a zkoušeny podle daného postupu, a to na platformách Linux nebo Windows. Je vybráno pět volně dostupných volných děl s různou délkou, která se stanou společným dělitelem všech zkoumaných lemmatizátorů. Stejná vstupní data umožní účelně **porovnávat** nejen kvalitu, ale i rychlost a formu výstupu. V rámci přípravy hodnotící škály pro srovnání kandidátů (nejvhodnějších nástrojů na základě požadavků) jsou zvolena **kritéria**, která odráží jejich budoucí použití, a poté budou porovnána mezi sebou.

Po vyhodnocení praktické stránky různých nástrojů jsou selektovány lemmatizátory splňující akceptační kritéria a jsou zařazeny jako **alternativa** do listu kandidátů ve zmiňovaném rozhodovacích softwaru. Kandidáti se utkávají v souboji o kvalitu, kvantitu, rychlost i snadnost použití. Jsou uvedeny konkrétní **výsledky** pokusů, které slouží jako podklad pro vyhodnocení nejlepšího rozhodnutí a umožní nad výslednými daty provést finální, citlivostní analýzu za podmínek autorovy subjektivity.

V diskuzi autor vyjádří své subjektivní postřehy, dedukce a doporučení z celého obsahu bakalářské práce.

2. Strojové zpracování textu

Pro strojové zpracování textu existují dva pojmenované koncepty.

Lemmatizace, což je proces, při kterém se každé slovo převede na svůj základní tvar a s**tematizace**, která hledá kořen slova. Uvedené potvrzuje [12, str. 2] „Lemmatizace hledá normalizovanou formu slova (*lemma*), kdežto během stemmingu jde o nalezení kořenu slova (*stem*)“.

Rozdíl v použití uvedených konceptů je podle [19] zřejmý: „Stemming se běžně používá např. pro angličtinu, ale u vysoce flektivních [Výraz 1] jazyků, jakým je čeština, je výhodnější použít lemmatizaci.“ Čeština vyjadřuje větnou syntax (skladbu) pomocí flexe. Vyznačuje se bohatstvím slovních tvarů ohebných slov [13].

Práce se pro uvedené důvody zabývá pouze procesem lemmatizace.

2.1. Lingvistické výrazy

Pro porozumění bakalářské práci považuje autor za potřebné mít znalost následujících doménových výrazů.

Výraz 1 - Flexe

Flexe (ohýbání) je modifikace tvarů slova (přesněji lexému), kterými se vyjadřují mluvnické kategorie, jako je např. pád, číslo, rod, čas, osoba atd. U slov, která se ohýbají, se vždy určitý tvar považuje za základní (lemma) [33].

Výraz 2 – Lemma

Dle [3] lze lemma definovat jako jednotku, která vzniká abstrakcí morfologických vlastností slovního tvaru. Lemma by mělo být základním lexikálním nositelem informace, nicméně je zdůrazňováno, že významnou část informace tvoří i tvar slova.

Podle Českého národního korpusu [3] platí, že

- „**lemma** každého českého substantiva je jeho nom. sg.[1] (tvary lesům, lesy, lesích má lemma les)“
- „u adjektiv je to nom. sg. mask. (tvary chytrého, chytrou, chytrejma má lemma chytrý)“
- „u sloves je to infinitiv (tvary „chodil“, „chodíš“, „chodíme“ má lemma chodit)“

Např. lemma slova „*jdeme*“ je „*jít*“. Původní slovo nese informace o morfologických vlastnostech, lemma tohoto slova již ale další informace nenese. K uchování informací o vlastnostech původního slova slouží **morfologické značky**.

Výraz 3 – Morfologická značka

„Morfologická značka (běžně nazývaná **tag**) je sumarizací gramatické informace o hledaném slovu (pozici) v konkrétním kontextu“ [3]. Např. tagset (soubor informací o jednom lexému) dle [3] užívaný v českých korpusech ČNK má 16 pozic a nese veškeré informace o tvaru daného slova. Zde je výčet všech 16 pozic.

1. Slovní druh - A = adjektivum (příd. jm.), V = sloveso, N = substantivum (podst. jméno), P = zájmeno, …
2. Detailní určení slovního druhu
3. Jmenný rod – F = femininum, M = maskulinum animatum (muž. životný), …
4. Číslo – P = plurál, S = singulár, …
5. Pád – 1. - 7. pád, x = lib. pád, - = neurčuje se
6. Přivlastňovací rod
7. Přivlastňovací číslo
8. Osoba
9. Čas
10. Stupeň
11. Negace
12. Aktivum/pasívum
13. Nepoužito

[1] Nominativ singulár (1. pád jednotného čísla)

14. Nepoužito
15. Varianta, stylový příznak apod.
16. Vid

Uvedený výčet šestnácti zkratek je tagsetem Českého národního korpusu. Významy daných zkratek slouží uživatelům ČNK (Český národní korpus) k orientaci v systému morfologických zkratek viz [31].

Nicméně existuje více druhů tagsetů, např. použité zkratky v softwarech *Majka*[40] nebo *MorphoDiTa* [38].

Pro uvedení příkladu s morfologickými značkami bylo využito morfologického desambiguátoru [30] [výraz 6], do kterého byla vložena věta „Karel jí uvolnil místo.“. Tabulka níže vypisuje výstup daného desambiguátoru.

token	Lemma	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
Karel	Karel	N	N	M	S	1	-	-	-	-	-	A	-	-	-	-	-
jí	ona	P	P	F	S	3	-	-	3	-	-	-	-	-	-	-	-
uvolnil	uvolnit	V	p	M	S	-	-	-	3	R	-	A	A	-	-	-	P
místo	místo	N	N	N	S	4	-	-	-	-	-	A	-	-	-	-	-
.	.	Z	:	-	-	-	-	-	-	-	-	A	-	-	-	-	-

Tabulka 1: Vlastní zpracování prostřednictvím formuláře [30]

Výstup z desambiguátoru je strojově zpracovatelný.

Pokud např. korektor českého jazyka v aplikaci MS Office Word používá podobný desambiguátor, snadno podle morfologických značek zjistí, že ve větě chybí např. sloveso.

Problematika mnohoznačnosti slov, která se v uvedeném příkladu vyskytuje, nás přenáší k dalšímu výrazu, a to k homografii.

Výraz 4 – Homografie

Homografie je dle [37, str. 10] slovo v takovém tvaru, u kterého lze bez ohledu na kontext vytvořit více základních tvarů. „Například tvaru ubrus lze jako základní tvar přiřadit nominativ substantiva ubrus nebo infinit slovesa ubrousit“ [37, str. 10].

V tabulce 1 výše si lze např. všimnout slova *jí*, které lze bez znalosti kontextu lemmatizovat na *ona* nebo *jíst*.

Výraz 5 – Homonymie

Všímavý čtenář se v tabulce 1 pozastaví i nad výrazem *místo*, které nese více významů, jako např. *místo něčeho* nebo *místo k sezení*. Tento jev je nazýván jako **Homonymie**.

„Nejjednodušeji můžeme homonymii charakterizovat jako jev, kdy jedna jazyková forma (např. lexém, morfém) je nositelkou dvou nebo více jazykových funkcí (významů)“ [5].

[3] uvádí jako příklad slovo *travička*, které v jednom významu může být tráva na louce a ve významu druhém žena, která někoho otrávila. [28, str. 128] uvádí lexém *jazyk* (1. svalnatý orgán v dutině ústní, 2. co se podobá jazyku živočichů, např. součást boty, 3. soustava vyjadřovacích prostředků) jako výskyt polysémie, neboť jednotlivé významy slova navzájem souvisejí, nebo si jinak podobají.

[28, str. 128-129] dále rozděluje a popisuje homonyma následovně.

- **Pravá homonyma** – lexémy, které mají různý význam a stejnou zvukovou i grafickou podobu.
- **Nepravá homonyma** – lexémy, které mají různý význam a
    - Homofona – stejnou zvukovou podobu a odlišnou podobu grafickou.
    - Homografa – odlišnou podobu zvukovou a stejnou podobu grafickou.

Výraz 6 – Morfologická desambiguace

Výskyt nejednoznačných případů, jako jsou homografie nebo homonymie, v procesu lemmatizování vede k potřebě tzv. desambiguace.

Morfologická desambiguace je proces, při kterém dochází k „odstraňování homonymie, čili jednoznačná interpretace slovního tvaru či skupiny slovních tvarů nebo věty na základě kontextu či mimojazykové situace. Desambiguace se obecně týká všech jazykových rovin, nejčastěji se ovšem v korpusech češtiny uplatňuje na rovině morfologické (zahrnující lemmatizaci a přiřazení náležitých morfologických údajů slovnímu tvaru na základě kontextu)“ [3].

[12, str. 6] jako příklad uvádí slovo *jí*, kde není význam slova jednoznačný. Při čtení ovšem člověk snadno podle kontextu pozná správný význam slova.

[12, str. 5] vysvětluje, že „princip morfologické desambiguace je založen na existenci vztahu mezi správným gramatickým významem vztahu aktuálně zpracovávaného slova a gramatickými významy slov v kontextu. Z tohoto kontextu je možné určit, jaký základní tvar slova je správný“.

Homografii a homonymii znázorňuje na příkladu i tabulka č. 2.

Jakýkoliv strojový algoritmus se ale musí o vztazích mezi správným gramatickým významem a gramatickými významy slov v kontextu nejdříve naučit, např. [12, str. 4] vytváří software s pracovním názvem Lemming, který se pravidla pro lemmatizaci učí podle vzorových textů. Zdrojem takových vzorových textů je např. **korpus**.

Výraz 7 – korpus

Korpus[2] je soubor textů, které jsou vnitřně strukturovány do různých celků. Korpusy psaného jazyka mají podobu třech sloupců. První obsahuje původní slovo, neboli token, ve druhém sloupci se nachází lemma daného slova a ve třetím sloupci najdeme morfologické značky [3], tzv. tagy, stejně jak je uvedeno výše v tabulce 1.

Rozdíl je u korpusů mluveného jazyka, kde se k daným slovům nevytváří lemma, ani morfologické značky, tj. korpus obsahuje pouze jediný sloupec.

2.2. Využití lemmatizace

Své využití lemmatizace nalézá ve **full-textových vyhledávačích**, jak popisuje Petr Strossa [39], kdy jedinec do vyhledávače zadá výraz „*sběrný dvůr*“, nicméně žádnou webovou stránku vyhledávač najít nemusí, protože v článcích o sběrných dvorech se sběrný dvůr nachází v jiných tvarech, např. *ve sběrném dvoru* nebo *před sběrným dvorem*. Stránky s tímto textem by mohly být pro čtenáře zajímavé.

Dalším nástrojem využívající lemmatizaci je **latentní sémantická analýza** (LSA). „Latentní sémantická analýza je technika, která zobrazuje dokumenty a dotazy do prostoru latentních

[2] „Český národní korpus je akademický projekt založený v roce 1994 při FF UK a spravovaný Ústavem Českého národního korpusu. Jeho cílem je systematicky mapovat češtinu a další jazyky ve srovnání s ní. Korpusy ČNK jsou po bezplatné registraci otevřeny všem zájemcům o jazyk, kteří touží vědět, jak se čeština používá“ [4].

sémantických dimenzí, přičemž slova, která jsou sémanticky podobná (měřeno mírou souvýskytů v dokumentech) jsou zobrazována do stejných dimenzí a slova sémanticky odlišná do různých dimenzí“ [19]. LSA pro každé slovo vytváří další dimenzi, dokumenty se tak mohou nacházet až v několika statisících dimenzí. Lemmatizace je zde vhodná z toho důvodu, aby počet slov zredukovala na minimum a to pomocí převedení všech slov na základní tvar. Tím se nevytváří různé dimenze pro stejná slova v jiném slovním tvaru. „Díky tomu mohou mít velkou sémantickou podobnost i dokumenty (případně dotaz a dokument), které spolu nesdílejí žádná slova“ [19]. LSA společně s lemmatizací nám umožní, abychom do vyhledavače zadali náš dotaz v jakémkoliv tvaru a našli webové stránky, které ani hledaný výraz neobsahují a plně vyhovují našim očekáváním.

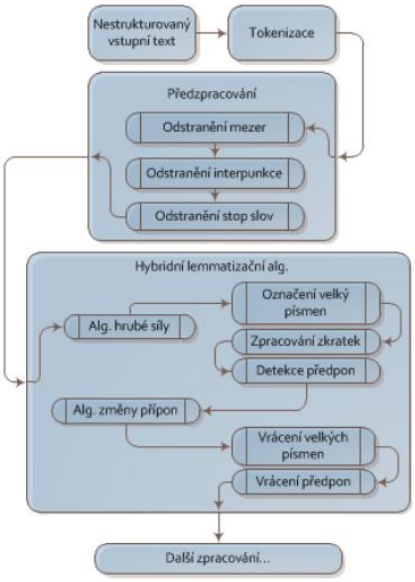
Automatická korektura textu. Např. společnost Lingea, s.r.o. se sídlem v ČR, nabízející překlady z ČJ do různých jazyků, na svých stránkách popisuje jazykové nástroje, které při své práci využívá či poskytuje. Jejími klienty jsou MF Dnes, Lidové noviny aj., kteří při své redakční práci využívají automatické dělení slov, kontrolu pravopisu a další. Společnost Lingea, s.r.o. taktéž nabízí korektor českého pravopisu, který podle Lingea, s.r.o. používáme v produktech Microsoft Office již od roku 1995 [9].

„Další oblastí, kde lze s výhodou využít automatického zpracování gramatických informací, je **korpusová lingvistika**“ [37, str. 7]. Lemmatizaci využívá software QUITA (Quantitative Index Text Analyzer) [20], který dokáže posuzovat a analyzovat rozsáhlé texty, např. bohatost slovní zásoby a další lingvistické ukazatele.

Lemmatizátory se nejčastěji vyskytují společně s tzv. tokenizátory a POS taggery, a tak tvoří jednu aplikaci označovanou obecně jako tzv. tagger.

2.3. Princip

Autoři v článku [12, str. 4-5] navrhují vlastní lemmatizační software s pracovním názvem Lemming (nejde o existující software Lemming [43] z Mnichovské Univerzity), jehož algoritmus je znázorněn obrázkem 1. Tento algoritmus je návrhem pro nezveřejněný software Lemming.



Obrázek 1: Průběh zpracování textu [12, str. 4]

Do algoritmu vstupuje hrubý nestrukturovaný text od uživatele, který je v dalším kroku **tokenizován**.

2.3.1. Tokenizace

Výsledkem procesu tokenizace je rozčlenění textu na **tokeny**, což jsou nejmenší jednotkou textu, většinou grafické slovo (tj. řetězec alfabetických znaků oddělených mezerou v textu)[3].

Například v aplikaci QUITA [20], kvantitativně lingvistický software, ukazuje Vladimír Matlach názornou implementaci metody „tokenizeText(ByVal sText As String)“, která jako parametr přijímá vstupní text a vrací pole tzv. tokenů. Lze tedy tokenizátor chápat jako to první, co přijde do styku se surovými daty od uživatele. V. Matlach taktéž v kódu aplikace rozlišuje tokenizátor vět a tokenizátor samotných slov ve větě. Podobně se zmiňuje taktéž o tom, že „chápání tokenů jako něco, co je od sebe odděleno mezerami, popř. jinými nealfanumerickými znaky, selže ve chvíli potřeby citlivějšího rozlišování, např. v angličtině *aren't* a *are not*. Podobně u vět selhává přístup **něco od tečky k tečce**“ [20, str. 8]. Slovo Karel IV. by podle přístupu „něco od tečky k tečce“ chybně rozdělilo jednu větu na dvě.

Konkrétní implementace záleží na požadavcích, které klademe. Pro nejjednodušší potřeby V. Matlach zveřejňuje jednoduchý tokenizátor zvaný „Default generic tokenizer“, který definuje token jako cokoliv, co je od jiného slova odděleno mezerou nebo jinými nealfanumerickými znaky.

Předtím, než text vstoupí do lemmatizátoru, je potřeba z textu **odstranit** mezery, znaménka interpunkce, popř. stopy slov, tedy taková slova, která nesou minimální význam. Příloha B v [12] vypisuje seznam všech takových slov, např. „(…), prosím, prostě, proti, protože, před, přes, přese, rovně, s, se, si, skoro, smí, snad, spolu, svůj, svých, ta, tady, (…)“.

2.3.2. Lemmatizace

Proces lemmatizace lze uskutečnit mnoha způsoby. Jeden z nich je představován jako algoritmus hrubé síly.

Algoritmy hrubé síly (brute-force algorithms) hledají normalizovanou podobu slova na principu prohledávání slovníku, který představuje tabulka, nebo jiná datová struktura) [12, str. 4].

Důvodem pro vznik **algoritmů hrubé síly** je flexe, stupňování přídavných jmen a jiné v češtině se objevující specifika, jako je sloveso *být* nebo přídavné jméno v různých stupních (dobrý, lepší, nejlepší). Systém na vyhledávání v tabulkách takové podobnosti najde a přiřadí vybraným slovům jejich základní tvar [12].

V článku [12, str. 4] je psáno, že lemmatizování **na základě slovníků není optimální řešení**, protože nejen že je téměř nemožné zachytit do slovníků všechna slova společně se všemi jejich tvary, neboť dochází k neustálému vytváření nových slov, ale jak Jan Karásek, Pavel Šanda, Radim Burget a Ondřej Morský [12] tvrdí: „problémem systémů založených na vyhledávání v tabulkách může být paměťová náročnost, jelikož je třeba udržovat několik velmi rozsáhlých tabulek, např. se slovními kořeny, slovními koncovkami, intersegmenty atd.“

Na základě výše uvedeného faktu, že lemmatizování na základě slovníků není optimálním řešením, se autoři [12] projektu s pracovním názvem Lemming rozhodli pro hybridní řešení. Například **zkratky** psané velkými písmeny se zcela vynechají z procesu lemmatizace (cca 10 tisíc) a pro zkratky, které nejsou jen z velkých písmen, tj. malá písmena, popř. kombinace malých a velkých písmen, se bude vést menší tabulka (cca o 2,5 tis. záznamech).

Na obrázku č. 1 v „bublince“ *detekce předpon* probíhá **odstraňování předpon** slov (jako ne-, nej-, po-) a zároveň se takovému slovu nastaví příznak, aby mohla být předpona později vrácena. Pomocí tabulky s výjimkami nedojde k odebrání těchto předpon u slov, kde se o předpony nejedná, např. jako slovo *nebesa*.

Petr Strossa [39] vidí lemmatizaci v porovnání s jinými jazyky **optimisticky**, citováno: „Mimochodem, víte, že čeština **má** vlastně ještě dost **štěstí**, že při lemmatizaci stačí (víceméně) pracovat s koncovkami? Je sice pravda, že to platí pro mnoho (možná většinu) jazyků světa, nicméně existují i takové, ve kterých může být na kmen slova přilepeno třeba deset gramatických přípon jedna za druhou (k těmto jazykům patří např. turečtina). A existují jazyky, ve kterých se slova ohýbají na obou koncích zároveň (do této skupiny patří např. irština nebo gruzínština)“.

Jak bylo zmíněno výše, pro rozlišení *homografie* [výraz 4] a *homonymie* [výraz 5] je potřeba morfologické desambiguace, která spočívá v odvozování na základě kontextu. Např. software LemmaGen [16] požaduje pro lemmatizaci jediné slovo, to znamená, že provádí **bezkontextovou** lemmatizaci.

V. Matlach [20] kritizuje bezkontextové dotazy na lemma jako nekvalitní, neboť analyzátor daným procesem ztrácí mnoho cenných informací, podobně jak zmiňuje [3], že význam je úzce spjat s morfologicky vymezeným tvarem.

Zautomatizovaný, naučený proces lemmatizace probíhá pomocí tzv. **Ripple-Down Rules**, které na rozdíl od slovníků hrubé síly používají strojově naučená pravidla.

Pro implementaci procesu lemmatizace se používá metoda strojového učení pravidel, tzv. **Ripple-Down Rules** [26] v kombinaci se slovníky, které řeší výjimky, jako jsou zkratky, vlastní jména anebo popř. idiomy.

Hlavní zásada **RDR** vypovídá o tom, že nejobecnější pravidla se stanovují jako první, a až poté lze přidávat různé výjimky (bloky „except“), které dané pravidlo upřesňují. RDR tak tvoří stromovou strukturu rozhodování. První podmínka, která je splněna a nemá žádné další výjimky, je uplatněna [10, str. 7]. Situaci znázorňuje následující obrázek č. 2.

IF bird THEN flies EXCEPT
    IF young bird THEN doesn't fly
    ELSE IF penguin THEN doesn't fly EXCEPT
        IF penguin in airplane THEN flies
ELSE IF airplane THEN flies

Obrázek 2: Struktura Ripple-Down Rules [10, str. 7]

Chování algoritmu na obrázku č. 2 lze upravit na základě přidání dalšího, specifického pravidla. Záměnou pravidel vytvořenými pravidly pro jiný jazyk lze použít stejný software na základě jiného jazykového modelu.

Poslední pravidlo na obrázku č. 3 společně s jeho výjimkou ukazuje rozšiřitelnost RDR. Například slovo *slůvka* se na základě stanoveného pravidla promění na základní tvar *slůvko*, nicméně slovo *borůvka*, které je svým způsobem výjimkou, již zůstane ve stejném tvaru.

```
RULE: i"vka" t""->"";
{:    RULE: i"ťovka" t"ťovka"->"tovka";
      RULE: i"ívka" t"a"->"o";
      {:    RULE: i"dívka" t"a"->"o";
            {:    RULE: i"#dívka" t""->"";
                  RULE: i"zdívka" t""->"";
            :}
            RULE: i"lívka" t""->"";
      :}
      RULE: i"ůvka" t"a"->"o";
      {:    RULE: i"růvka" t""->"";
      :}
:}
```

Obrázek 3: Část kódu Ripple-Down Rules - LemmaGen [11]

Pro vytvoření Ripple-Down Rules potřebuje generátor trojici slov, a to slovo v libovolném slovním tvaru, lemma a jeho morfologické značky.

Algoritmus, který vytváří RDR, na svůj vstup potřebuje trojice, tj. lexém v libovolném tvaru, lemma a morfologické značky. Pokud zaměníme lemma v každé trojici za stemma, získáme místo lemmatizátoru stemmatizátor [10, str. 6].

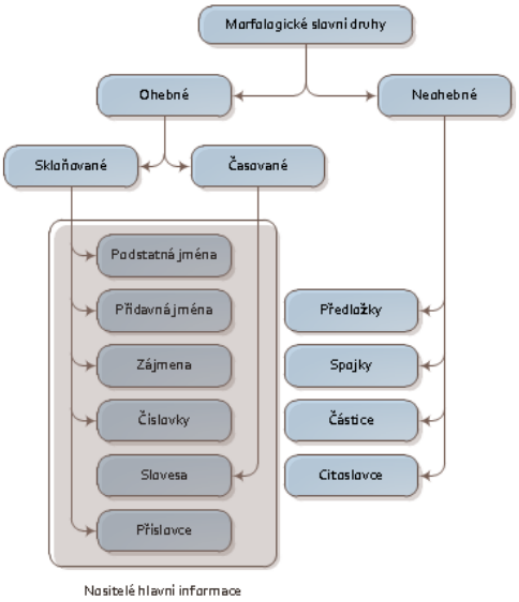
Tento předpoklad splňuje **Český národní korpus** [Výraz 7], který odpovídá požadované struktuře.

Metodu Ripple Down Rules je výhodné používat v kombinaci s algoritmy hrubé síly, kdy v tabulkách uchováváme pouze záznamy různých nepravidelných slov, vlastních jmen, popř. zkratek a dalších specifik [12, str. 4].

2.3.3. Part-Of-Speech tagging

POS tagging je proces přiřazování morfologických značek každému tokenu.

Na obrázku č. 4 lze vidět morfologické rozdělení slovních druhů. Důležité je si uvědomit, které slovní druhy jsou skutečnými nositeli informace. Někdy i zdánlivě nevýznamná slova mohou totiž rozhodnout o významu věty. Dalo by se tak nesouhlasit s výrokem, že „neohebné slovní druhy nenesou žádnou nebo jen minimální informaci a bývají při zpracování textu odstraněny“ [12, str. 3].



Obrázek 4: Morfologicdké rozdělení slovních druhů [12, str. 3]

Nejdříve proběhne morfologická analýza, která ke slovu vytvoří až několik lemmat s danými morfologickými značkami, jak ukazuje následující tabulka č. 2.

Token	Lemma	Tag
Sním	**sníst**	**VpS--1d**
	snít	VpS--1n
je	být	VpS--3n
	ono	PPSN4--
	oni	PPPM4--
	ony	**PPPI4--**
	ony	PPPF4--
	ona	PPPN4--
místo	místo	NNSN1--

	místo	**R---2--**
něho	něha	NNSF5--
	on	**PPSM2--**
	on	PPSM4--
	on	PPSI2--
	ono	PPSN2--
.	.	Z------

Tabulka 2: Morfologická analýza věty „Sním je místo něho“ [3]

Po práci morfologického analyzátoru vstupuje morfologický desambiguátor, který operuje na výsledcích morfologického analyzátoru. „Výsledkem správné lemmatizace a morfologické desambiguace následující po morfologické analýze uvedené věty jsou interpretace označené tučně“ [3].

2.4. Specifika při lemmatizaci

Obtížným specifikem jsou **víceslovná spojení**, tj. vytváření lemmat i tam, kde to není možné, např. zdvořilá prosba *Dovolíte?* se nenachází v žádném z registrovaných významů slova dovolit, dále se může jednat o frazémy, např. *nechat na holičkách*, popř. se jedná o idiomy např. *z někoho si vystřelit* [3].

Homografie a homonymie jsou pojmy, s kterými si algoritmy hrubé síly neporadí tak jako s vlastními jmény a s názvy obcí.

2.5. Datový formát csts

Datový formát csts (Czech sentence tree structure) je založen na SGML [23] (Standard Generalized Markup Language), což je dle Jiřího Koska [14] „standardní jazyk určený k formálnímu popisu struktury dokumentů. Je definován v normě ISO 8879 z roku 1986“.

„Standard SGML vznikl v rámci projektu ODA (Open Document Architecture)“. Z uvedeného volně vyplývá, že jde o formát pro ukládání strukturovaných dat, které mají být díky formátu snadno přenositelné i napříč různými platformami.

SGML je *metajazyk*, pomocí kterého lze definovat značky. Nové formáty založené na SGML se definují pomocí DTD (Document Type Definition).

Další aplikací standardu SGML pomocí DTD souborů je i HTML nebo RDF. Jako výhodu SGML standardu Jiří Kosek označuje jeho otevřenost a zároveň definitivnost (myšleno v kontextu vytvoření nové verze standardu).

„Pokud je DTD dobře navrženo, je velmi usnadněna i automatická syntéza řeči (tagy vyznačují místa, kde se obvykle i v řeči dělají pauzy), což je velkým pozitivem při zpřístupňování informací nevidomým občanům“ [14].

Podle [23] vypadá datový formát CSTS na příkladu takto:

<f>slovníTvar<l>ručníLemma<t>ručníZnačka<MMl>lemmaZMorfAn1<MMt>značkaZMorfAn1...

„Soubor CSTS začíná (nepovinnou) hlavičkou (element *h*) a dále obsahuje alespoň jeden element doc. Element doc sestává z hlavičky (element *a*) a obsahu (element *c*). Element *c* pak sestává z posloupnosti odstavců (element *p*) a vět v těchto odstavcích (element *s*)“ [2].

2.6. Dostupné software

2.6.1. Ajka

Ajka je software, který vzešel z diplomové práce Radka Sedláčka na Masarykově Univerzitě na Fakultě Informatiky s názvem „Morfologický analyzátor češtiny“ v roce 1999. O Ajce je zmíněno v publikaci [27] v kontextu použití strojového slovníku, který se stal bází softwaru *Ajka*, obsahující 170 000 kmenů, z nichž každý kmen má přiřazeno pravidlo, které určuje trojici lemma, generovaný slovní tvar a tagset.

Online rozhraní je ovšem omezeno na 100 znaků, nebo na velikost 2kb nahraného textového souboru.

Software není na internetu dostupný ke stažení. Budeme se zabývat jeho nástupcem, který ho nahradil, a to *Majkou*.

2.6.2. Majka

Majka [40] plynule pokračuje na předchozí zmíněný software zvaný *Ajka*, jejími autory jsou Pavel Šmerk a Pavel Rychlý.

K dispozici binární soubory pro Linux a pro Windows. Majku lze s úspěchem použít na obou operačních systémech prostřednictvím příkazové řádky.

Stejně jako Ajka, tak i Majka pochází z Masarykovy univerzity Fakulty informatiky. Nabízí možnost lemmatizace a POS taggingu v 15 různých jazycích. Na druhou stranu neposkytuje tokenizaci a všechen svůj vstupní text si uživatel musí na slova rozdělit sám.

Majka požaduje na vstup jedno jediné slovo a výstupem je více možných reprezentací daného slova s morfologickými značkami.

Majka je šířena pod licencí GLP verze 2 a její česká morfologická databáze pod licencí Creative Commons Attribution-ShareAlike 3.0, což umožňuje Majku, resp. její český datový model, používat neomezeně pouze pro nekomerční účely.

2.6.3. MORČE

MORČE [35] (zkratka „MORfologie Češtiny“) vyvinul Jan Raab na ÚFALu.

Dle [21] je MORČE morfologický desambiguátor přesný na 95%. Morče v sobě zahrnuje jak tokenizér, lemmatizér tak i tagger. Morče určuje správné lemma a morfologické značky podle kontextu.

Morče vytváří pro jedno slovo všechna možná lemmata a poté na základě statistického algoritmu rozhoduje, které z daných lemmat, popř. morfologických tagsetů, je nejvhodnější.

Lemmatizér byl dle popisu projektu trénován na PDT 2.0[3]. Přesnost této verze je tedy po stovkách experimentů hledající nejlepší kombinaci funkcí přesný na 95,1%. Pro registrované uživatele jsou k dispozici funkce zaručující úspěšnost až 96,0%.

Prohlášení na webových stránkách tohoto morfologického taggeru “Developement of Morče is supported from grant 1ET101120503 of Academy of Sciences of the Czech Republic and grant GD201/05/H014 of Czech Science Foundation.“ říká, že vývoj je podporovaný z grantu pro

[3] The Prague Dependency Treebank 2.0 (PDT 2.0) – je soubor velkého množství textů s komplexní, propojenou morfologickou analýzou a se sémantickými anotacemi [41]

integraci jazykových zdrojů za účelem extrakce informací z přirozených textů [32], z kterého bylo čerpáno mezi roky 2005 a 2009.

2.6.4. LemmaGen

LemmaGen [16] byl původně určen pouze pro lemmatizaci **slovinštiny**, nyní s pomocí strojového učení podporuje dalších 14 evropských jazyků (včetně češtiny). LemmaGen je dostupný pod licencí open source a je k dispozici v programovacím jazyku C++, C++.Net, C++ pro Python a C#.Net. Vytvořené funkční jazykové modely ale součástí open source nejsou, ale i tak jdou použít volně pro nekomerční účely.

LemmaGen je pouze lemmatizátorem, tzn., že rozdělení souvislých vět na jednotlivá slova zůstává na uživateli, podobně jako určování morfologických značek.

Dle slov autorů LemmaGenu může být projekt překompilován pro Windows nebo Linux. Lemmatizátoru nezáleží na struktuře věty, protože je lemmatizace aplikována na každé slovo zvlášť. Autoři vyzdvihují schopnost lemmatizátoru zvládnout až **milion slov za vteřinu**, dobrou dokumentaci a podporu. Projekt podkládají potvrzenou kvalitou jazykových modelů [16].

Autoři tohoto projektu slovinského Institutu Jožef Stefan jsou Matjaz Jursicze za vývoj, Nada Lavrac za koncept a Mozetic Igor, Tomaz Erjavec, Miha Grcar, Vid Podpecan jako další podpora. Všichni zmínění jsou zodpovědní za vznik tohoto software a nabízejí své odpovědi na případné otázky a požadavky k tomuto projektu.

Nezávisle na autorech LemmaGenu vznikl spontánně projekt JLemmaGen, u něhož již název vypovídá, že se jedná o přepsanou verzi LemmaGen do javy.

LemmaGen je jeden z mnoha projektů tohoto institutu.

2.6.5. MorphoDiTa

MorphoDiTa [38] je dalším dílem ÚFALu, autorů Milana Straky, Jany Strakové a dalších.

Již na samotném webovém rozhraní projektu je MorphoDiTa nabídnuta k odzkoušení, v neomezeném množství a v libovolném nastavení. Je k dispozici také REST service, které je dostupné přes veřejné API.

MorphoDiTa je šířena pod licencí Mozilla Public License 2.0 a jeho lingvistické datové modely pod licencí CC BY-NC-SA, což umožňuje použití tohoto software v neomezeném množství pro nekomerční použití.

Dle [24] dosahuje MorphoDiTa rychlosti mezi 10 až 200 tisíci slovy za vteřinu. Tento nástroj si uživatel spustí jako samostatný program, zapojí do projektu nebo na svém stroji spustí REST server. Projekt je k dispozici pro platformy Linux, Windows nebo OS X, buď jako samostatná aplikace, nebo jako knihovna použitelná v projektech. Software nevyžaduje žádné další knihovny, kromě natrénovaných lingvistických modelů.

Prohlášení: „This work has been using language resources developed and/or stored and/or distributed by the LINDAT/CLARIN project of the Ministry of Education of the Czech Republic (project LM2010013).“ Na projekt se jménem MorphoDiTa byly čerpány prostředky ze státního rozpočtu na projekt s označením LM2010013 [24].

Pro software je dostupná rozsáhlá dokumentace ve formátu pdf. Nástroj je dále udržovaný a stále se rozvíjí.

2.6.6. Cistern

Cistern je označení pro rozsáhlou sadu nástrojů pro zpracování textu, které byly realizovány v „Center for Information and Language Processing“ na Mnichovské univerzitě. Mezi tyto nástroje patří např. Lemming (lemmatizér) a MarMoT (morfologický tagger), o kterých je zmínka dále. Na webových stránkách tento projekt prezentují jako open source nástroje pro morfologické tagování, zpracování textu a další nástroje pro NLP (natural language processing) [43].

2.6.6.1. Lemming

Kontaktní osobou a stvůrcem tohoto software je Thomas Müller. Lemming je statistický lemmatizer, ke své činnosti potřebuje part-of-speech informace, které získá při spolupráci s programem MarMoT [17]. Jde o software napsaný v javě. Na vstup vyžaduje zpracovaný text v takovém formátu, aby byla slova oddělená odřádkováním a věty potom řádkem prázdným.

Trénování nových datových modelů je podle [17] ve špatném stavu, nicméně stále se dá použít podle ukázkového příkladu na uvedených webových stránkách.

2.6.6.2. MarMoT

Podobně jako Lemming, který je tímto softwarem vyžadován, běží i MarMoT v prostředí javy, je tak spustitelný na všech platformách s podporou JVM. Autorem je Thomas Müller.

Na oficiálních stránkách projektu [18] uživatel nalezne odkazy na dokumentaci, zdrojové kódy, poslední spustitelnou verzi projektu a natrénované jazykové modely, které jsou, stejně jako v ostatních softwarech, volně dostupné pouze pro nekomerční použití.

Software je šiřitelný pod licencí GNU General Public License v. 3.

2.6.7. Czech HMM tagger

Software z řady nástrojů vytvořených na ÚFALu. Autorem je Pavel Krbec (HMM tagger) 2001 a Jan Hajič (morphology) 2001. Czech HMM-based Tagger (using full morphology) je k dispozici ke stažení pro operační systém Linux. Ke své činnosti tento tagger vyžaduje nástroj Czech „Free“ Morphology [15].

Vstupními data musí následovat kódování ISO Latin 2 (iso-8859-2) a být ve formátu csts. Výstupem jsou data ve stejném formátu jako data vstupní.

Jako tip pro dosáhnutí většího výkonu, [15] doporučuje mít složku /tmp na lokálním úložišti a pokud uživatel tagguje více menších souborů, radí spojení těchto souborů do jednoho, neboť se natrénovaná data znovu nahrávají s každým požadavkem, což má tendenci být výpočetně náročné.

Czech HMM tagger konzumuje mezi 13 až 20 MB paměti. Čím více paměti dostane, tím jsou podle [15] vstupně výstupní operace rychlejší.

2.6.8. Czech "Free" Morphology

Free Morphology je pár univerzálních nástrojů pro analýzu a generování slovních tvarů pro flektivní jazyky [6]. Autorem je Jan Hajič, 2000 - 2001.

Vstup do softwaru Free Morphology je „free-format text (in either ISO Latin 2 or MS Windows CP 1250 8-bit coding)“[6]. Software také rozpozná csts format, html format nebo jinou aplikaci standardu SGML. Program tokenizuje vstup na slova, nikoliv ale na věty.

Jako příklad se uvádí zpracování věty „Prezident rezignoval na svou funkci“ a uvádí se následující výstup:

```
<csts>
      <f cap>Prezident<MMl>prezident<MMt>NNMS1-----A----
      <f>rezignoval<MMl>rezignovat_:T<MMt>VpYS---XR-AA---
      <f>na<MMl>na<MMt>RR--4----------<MMt>RR--6----------
      <f>svou<MMl>svůj-1_^(přivlast.)<MMt>P8FS4---------1<MMt>P8FS7--
      -------1
      <f>funkci<MMl>funkce<MMt>NNFS3-----A----<MMt>NNFS4-----A----
      <MMt>NNFS6-----A----
      <D>
      <d>.<MMl>.<MMt>Z:-------------
</csts>
```

2.7.Další zajímavé nástroje

2.7.1. QUITA

QUITA [20] (Quantitative Indicator Text Analyzer) je výsledkem diplomové práce Vladimíra Matlacha při studiu na Filozofické fakultě Univerzity Palackého v Olomouci na Katedře obecné lingvistiky. QUITA je spustitelná pouze na platformě Windows, obsahuje spoustu nástrojů pro analyzování textů a disponuje intuitivním grafickým uživatelským rozhraním.

Software využívá vlastní tokenizátor, nicméně k lemmatizaci samotné a POS taggingu využívá služeb jiných knihoven. Pro český jazyk si tak můžeme zvolit lemmatizátor Majka ve spolupráci s Českým národním korpusem.

QUITA je distribuovaná jako freeware.

2.7.2. Morfo

Morfo, český morfologický analyzátor, vznikl taktéž na půdě ÚFALu, autorem je David Kolovratník a Leoš Přikryl.

Software plní funkci morfologického analyzátoru. Jednomu slovu na vstupu přiřadí seznam možných lemmat a tagů. Umí také podle zadaného lemmatu vytvořit veškeré slovní tvary, tedy všechny, které je schopné podle svého slovníku udělat. Pro spravování morfologického slovníku, kde mohou být vloženy další záznamy, nabízí grafické uživatelské rozhraní [22].

Lemmatizátor neplní funkci tokenizátoru a daným slovům nepřiřazuje jednoznačné lemma, ale plný výčet všech možných lemmat a tagů, podobně jako Majka.

Morfo je šířen pod licencí GNU compatible (libPNG) a použití je omezené pouze na platformu Linux. Projekt obsahuje obsáhlou dokumentaci.

2.7.3. Compost Czech

Morfologický **tagger** z řady projektů ÚFALu, který kombinuje PDT 2.0 a tagger Morče. Je k dispozici pro 5 světových jazyků (angličtina, čeština, holandština, švédština, islandština). Podle [34] dosahuje přesnosti určování tagů až 96%.

Compost Czech je dostupný pouze pro platformu Linux.

Podle dokumentace ve stáhnutém programu v souboru README je návod k použití. Návod k použití je *./compost_cz.sh <input> <output>*. Na vstup přijímá text s každým slovem na samostatném řádku a věty oddělené řádkem prázdným. Výstupem je stejný text obohacený o morfologické značky, např. *Hrad: NNIS1-----A----*.

2.7.4. RDRPOSTagger

Robustní **tagger** (Ripple Down Rules Part-Of-Speech Tagger), napsaný v Pythonu, poskytuje natrénované jazykové modely pro 40 různých jazyků. Autory jsou Dat Quoc Nguyen, Dai Quoc Nguyen, Dang Duc Pham a Son Bao Pham [25]

Poslední aktualizace softwaru je 17. října 2016. Rychlost zpracování dosahuje až 90 tisíc slov za vteřinu vzhledem k tomu, že se jedná o jedno-vláknovou aplikaci na stroji s Core2Duo 2.4GHz a 3GB paměti [25].

2.7.5. Ant

Dr. Laurence Anthony, profesor na Univerzitě Waseda, Faculty of Science and Engineering v Japonsku, přichází s užitečnými nástroji pro zpracování asijských nebo evropských textů [1].

2.7.6. MorphCon

MorphCon, nástroj pro automatickou konverzi morfologických taggovacích sad, vznikl v roce 2008 za spolupráce tří univerzitních pracovišť:

- Katedra bohemistiky FF UP Olomouc
- Fakulta informatiky Univerzity Bonn (Německo)
- Ústav formální a aplikované lingvistiky MFF UK Praha [29]

Jejími hlavními představiteli jsou Petr Pořízka, Marek Schäfer a Daniel Zeman

2.7.7. Interset

Interset, který se věnuje automatické konverzi morfologických taggovací sad, je nástrojem ÚFALu, napsaný v jazyce Perl, umožňuje pokročilým uživatelům pomocí přiložené dokumentace napsat si vlastní ovladač pro převod z jedné sady morfologických značek do jiné [7].

3. Analýza

Veškeré postupy i metody byly provedeny na výpočetní technice ASUS s Intel Core i5-4200U 1.60GHz * 2 s operační pamětí 8GB, SSD diskem a s grafickou kartou NVIDIA Geforce 840M 2GB. Rychlost zkoumaných softwarů může být, a pravděpodobně také i bude, odlišná na jiné výpočetní technice.

3.1. Výběr vzorových textů

Pro ozkoušení vybraných softwarů bylo vybráno pět různě dlouhých textů volných děl [42].

Volné dílo	Znaky bez mezer	Slova
Jan Neruda – Povídky Malostranské (1. kapitola V Košili)	5 403	1 103
Karel Čapek – Povídky z jedné kapsy	225 043	46 543
Alois Jirásek – Staré pověsti české	400 485	82 371
Božena Němcová – Babička	348 037	72 885
Bible – ekumenický překlad	2 770 660	595 410

Tabulka 3: Volná díla a jejich délka ve znacích a slovech [vlastní zpracování]

3.2. Tokenizace textu

Pro případ, kdy v sobě zkoumaný nástroj neobsahuje tokenizátor, tj. takový nástroj, který rozdělí souvislý text na slova, popř. na věty, byl v práci použit nástroj od autora Andrew Roberts **JTokenizer** verze 2.0 implementovaný v Javě [36].

Po předchozí zkušenosti s většinou nástrojů je známo, že většina software vyžaduje na vstup textový soubor, který obsahuje slova oddělená odřádkováním a věty oddělené prázdným řádkem.

Převod textů do této podoby byl v JTokenizeru proveden ve dvou krocích.

1. Do vstupního pole v grafickém uživatelském rozhraní JTokenizeru byl vložen nijak nezpracovaný text, vybrán „SentenceTokeniser“, Locale nastaven na češtinu, Token prefix (vložený znak před tokenizovaný výstup) nastaven na prázdnou hodnotu a Token suffix

(vložený znak za tokenizovaný výstup) nastaven na „\n\n“[4]. Toto nastavení při kliknutí na tlačítko „Tokenise“ rozpoznané věty od sebe oddělí dvojitým odřádkováním, tj. vytvoří mezi větami jeden prázdný řádek.

2. V kroku druhém byl předchozí výstup použit jako vstup, ovšem nyní byl Tokeniser nastaven na StringTokeniser s Delimiterem „ \t\r\f“, Token prefix zůstal prázdný a Token suffix byl nastaven na „\n“. Nyní všechna slova oddělí odřádkováním a mezi větami zůstane prázdný řádek.

Všechny texty kromě Bible se podařilo ztokenizovat do pěti minut. Bible, která je svým obsahem tak osmkrát rozsáhlejší než ostatní, byla na daném stroji zpracována za půl hodiny.

3.3. Stanovení kritérií

Je stanoveno nekomerční prostředí s vědeckými účely, konkrétně Fakulta informatiky a managementu Univerzity Hradec Králové, pro které tato práce podle daných požadavků vyhledá a stanoví nejvhodnější analytický software z výběru dostupných lemmatizátorů. Jedním z logických požadavků je volné nekomerční použití, aby bylo možné neomezeně využívat technologie pro různě obsáhlé výzkumy. Aby bylo zajištěno, že software bude možné používat delší dobu, je žádoucí vybrat pravidelně aktualizovaný a dále vyvíjející se software.

Kritéria byla stanovena následovně:

1. Licence – Volně dostupné pro nekomerční použití
2. Platforma – Nejlépe s podporou pro Windows
3. Podpora – Stále podporovaný vývoj
4. Práce s kontextem – Kvalita lemmatizace a taggeru
5. Rozsah textu a rychlost – Kvantita
6. Snadnost použití – Jednoduché rozhraní
7. Vhodnost pozičních tagů – Počet tagů a jejich význam

[4] Znak „\n“ symbolizuje odřádkování

Licence je u všech zkoumaných softwarů podobná, tj. neomezeno pro nekomerční použití. Zařazení tohoto kritéria do rozhodování by nebylo účelné, neboť by se tím jednotlivé softwary nedaly porovnávat.

3.4. Software pro podporu rozhodování Expert Choice

Pro porovnání dostupných softwarů pro lemmatizaci byl použit Expert Choice 2000 Team 10.1 Build 903.05. Protože software není volně dostupný, byla využita licence udělená FIM UHK a bylo s ním pracováno na virtuálních učebnách Univerzity Hradec Králové Fakulty informatiky a managementu.

Ze seznamu dostupných lemmatizátorů budou vybráni **potencionální** kandidáti, kteří budou prostřednictvím citlivostní analýzy v softwaru Expert Choice hodnoceni, a ten s nejvyšším hodnocením bude Fakultě informatiky a managementu doporučen jako nejvhodnější.

Nejdříve byl stanoven tzv. „goal“, tedy cíl, kterého chce uživatel dosáhnout. Cíl byl nazván jako „Nalézt nejvhodnější lemmatizér“.

Poté bylo nutné zadat do softwaru Expert Choice kritéria, podle kterých se budeme rozhodovat. Byla využita kritéria stanovená výše. Konkrétně to je platforma, podpora, kvalita, kvantita, snadnost použití a vhodnost pozičních tagů.

Po zadání hodnotících kritérií následuje určení vah jednotlivých kritérií. K tomu slouží přehledná tabulka, v které se porovnává významnost jednoho kritéria s jiným.

	Platforma	Podpora	Práce s k	Rozsah te	Snadnost p	Vhodnost
Platforma		4,0	5,0	3,0	3,0	4,0
Podpora			3,0	3,0	2,0	4,0
Práce s kontextem - Kvalita				4,0	3,0	5,0
Rozsah textu a rychlost zpracování - Kvantita					3,0	3,0
Snadnost použití						3,0
Vhodnost pozičních tagů	Incon: 0,09					

Obrázek 5: Porovnávací tabulka v nástroji Expert Choice [vlastní zpracování]

Hodnocení jednotlivých kritérií je silně subjektivní záležitostí a záleží na cítění jedince. Autorovu volbu znázorňuje obrázek výše. Červeně označená čísla v tabulce symbolizují „plusové body“ pro kritérium v záhlaví tabulky, černě obarvená čísla pak přidávají významnosti kritériím v prvním sloupečku tabulky.

Modré pozadí za kritérii v prvním sloupečku potom znázorňují rozložení vah jednotlivých požadavků. V poslední fázi, tj. v citlivostní analýze, bude možné tyto váhy ovlivňovat.

Hodnota *incon* říká, do jaké míry je naše porovnání kritérií konzistentní. Nekonzistentním je takové porovnání, kdy říkáme, že X je lepší než Y, Y je lepší než Z a Z je lepší než X. Hodnota této automaticky vypočítané hodnoty by měla být v intervalu od nuly do jedné desetiny.

3.5. Postup zkoušení lemmatizátorů

Vzhledem k tomu, že každý nástroj poskytuje rozdílné rozhraní, je potřeba ke každému zvlášť přistupovat individuálně. Některé operují pouze na operačním systému Linux, některé pouze na Windows a některé zvládají obojí. Pro tento případ jsou připravené obě prostředí. V prvním případě se jedná o Linux Mint 18 Cinnamon 64-bit a v případě druhém Windows 7 64-bit.

Všechny vybrané nástroje projdou obecně těmito kroky

1. Nainstalování/zprovoznění nástroje
2. Prozkoumání všech možností softwaru
3. Lemmatizace a označkování vzorových textů
4. Vyzkoušení zlemmatizování vzorových textů a změření jejich rychlosti
5. Zhodnocení výkonosti a použitelnosti

3.6. Vyzkoušení nástrojů

3.6.1. Majka

Majku lze stáhnout včetně datových modelů z webových stránek Masarykovy univerzity Fakulty informatiky. Pro ozkoušení Majky byl zvolen binární soubor pro operační systém Linux.

Použití tohoto lemmatizátoru je snadné. Pro získání základního tvaru libovolného slova, např. *oni*, stačí zadat do příkazové řádky v daném adresáři s binárním souborem *majka* a s datovým souborem *.w-lt tento vstup.

```
echo oni | ./majka -f model.w-lt
```

Na Linuxu je potřeba zadat **./majka** namísto **majka.exe**. Argument –f je cestou k souboru s daty, podle kterých Majka lemmatizuje a určuje tagset. Snadno se tak dá lemmatizovat podle patnácti rozdílných jazyků pouhou záměnnou datových souborů. Výstupem tohoto konkrétního případu v českém jazyce je následující analýza:

```
on:k3p3gMnPc1xP
onen:k3gMnPc1xD
```

Výstup programu je snadno strojově zpracovatelný, neboť samotné lemma a jeho tagset jsou oddělené dvojtečkou. Pro převod uvedeného tagsetu do jiných tagsetů by bylo možné využít překladače MorphCon nebo Interset.

Majka v sobě neobsahuje tokenizér, nepodporuje kontextovou desambiguaci.

Majka nabízí další dva nástroje pouhou záměnou datových modelů.

- V příkladu výše je použit datový model s příponou **w-lt**, který přiřazuje danému slovu všechna možná lemmata a tagy.
- Datový model s příponou **l-wt** přiřazuje pro zadané lemma všechny slovní tvary společně s tagy. Např. *echo hrad | ./majka -f model.l-wt*

```
hrad:k1gInSc1
hrad:k1gInSc4
hrade:k1gInSc5
hradech:k1gInPc6
hradem:k1gInSc7
hradu:k1gInSc2
hradu:k1gInSc3
hradu:k1gInSc6
hrady:k1gInPc1
hrady:k1gInPc4
hrady:k1gInPc5
hrady:k1gInPc7
hradě:k1gInSc6
hradů:k1gInPc2
hradům:k1gInPc3
```

- Třetí model s příponou souboru **lt-w** generuje slovní tvary na základě zadaného lemma. echo hrad | ./majka -f model.lt-w

```
k1gInPc1:hrady
k1gInPc2:hradů
k1gInPc3:hradům
k1gInPc4:hrady
k1gInPc5:hrady
k1gInPc6:hradech
k1gInPc7:hrady
```

```
k1gInSc1:hrad
k1gInSc2:hradu
k1gInSc3:hradu
k1gInSc4:hrad
k1gInSc5:hrade
k1gInSc6:hradu
k1gInSc6:hradě
k1gInSc7:hradem
```

Majka nabízí další rozhraní, a to např. agrument *-p*, kterým dostaneme naše vstupní slovo taktéž do výstupu a zároveň bude výstup na jednom řádku, argument *-d,* který přidá diakritiku a argumenty *-i* jako *ignore case* a *-l* jako *do NOT lowercase*, které si hrají s nastavením velikosti písmen.

Majka nenabízí způsob, jak pro jeden vstup získat pouze jeden výstup.

3.6.2. MORČE

Pro použití MORČEte stačí zadat do příkazové řádky následující řádek:

morce_run.bat inputfile.csts outputfile.csts

Vstupní soubor musí být ve formátu csts. Zlemmatizovaný a otaggovaný výstup nalezneme taktéž ve formátu csts. Díky jeho formalizované podobě jde o snadno strojově zpracovatelný výstup. Pro ozkoušení lemmatizátoru se nepovedlo vytvořit, ani sehnat dostupný textový soubor ve formátu csts.

3.6.3. LemmaGen

Podle oficiálních stránek projektu LemmaGen [16] není verze 3.0 (Pro C#) a 2.5 (Python wrapper for .NET) samostatnou aplikací pro použití v příkazové řádce, ale plní účel knihovny importovatelné do jiných projektů. V příkazové řádce lze použít předchozí verze, tj. 1.0, 1.5, 2.0 a 2.2, které jsou v C++ a navazují na sebe vývojem. Protože nejnovější verze 2.2 není k datu 26. 3. 2017 dostupná, bude použita verze **2.0**, dostupná v binární podobě pro Linux i Windows.

Pro spuštění byl v příkazové řádce zvolen soubor *lemmagen*, který obaluje podle následujícího výstupu ostatní funkcionalitu a který po zadání příkazu *./lemmagen –h* vykreslí následující:

```
Wrapper of complete lemmagen library funcionalty.
Usage: lemmagen <subproc> [<subproc switches>]

  <subproc>     specify subprocess to run, possible choiches:
```

```
     lemLearn    Learns RDR tree from examples in multext format
     lemBuild    Builds lemmatizer data from the rdr tree def. file
     lemmatize   Lemamtizes text and produces new lemmatized file
     lemXval     Validates accurancy of learning algorithm
     lemTest     Calculates lemmatization accurancy from two files
     lemStat     Calculates statistics about input example data file
     lemSplit    Splits example file into subsets for validation

  <subproc switches>     try help of the subprocesses for switches
definition

  -h, --help             print this help and exit
      --version          print version information and exit
```

Lemmagen umožňuje naučení se nových pravidel podle zadaného vstupu, vybudovat nový lemmatizer na základě dat podle RDR stromu a další.

Podle nápovědy lze program *lemmatize* použít následovně:

```
Lemamtizes text and produces new lemmatized file.
Usage: lemmatize [-f <format>] [-l <datafile>] [-lm <lm>] <infile>
<outfile>
```

Podle vypsaného návodu autor práce použil tento příkaz pro ozkoušení lemmatizace.

```
./lemmatize -f text -l ../../../data/lemmatizer/lem-me-cs.bin -lm
b ./input.txt ./output.txt
```

Argument *–f* vypovídá o vstupním souboru, na výběr je mezi hodnotou *text* a *wpl* (word per line). Argumentem *–l* je cesta k souboru s RDR, *-lm* argument má na výběr ze tří hodnot, a to *binary* (binary data file – optimized lemmatization), *text* (textual data file – NOT optimized lemmatization) nebo *optimtext* (textual data file – optimized lemmatization). Nakonec následuje vstupní a výstupní soubor.

Výstupem vstupu „*Sním je místo něho.“* v tomto softwaru je „*Sní on místo on.*“. Můžeme použít taktéž argument *–d* s libovolnou hodnotou. Tento argument na výstupním souboru všechna slova oddělí námi definovanou hodnotou.

3.6.4. MorphoDiTa

Nejdříve si musí uživatel stáhnout binární soubory pro náš operační systém a poté datový balíček, který je pro všechny operační systémy stejným souborem. Po zadání příkazu ./**run_tagger --help** se vypíše nápověda pro použití software.

```
Usage:  ./run_tagger [options] tagger_file [file[:output_file]]...
```

```
Options:   --input=untokenized|vertical
           --convert_tagset=pdt_to_conll2009|strip_lemma_comment|
           strip_lemma_id
           --derivation=none|root|path|tree
           --guesser=0|1 (should morphological guesser be used)
           --output=vertical|xml
           --version
           --help
```

Činnost softwaru MorphoDiTa je tak do možností nastavení pod uživalskou kontrolou. Hodnoty parametrů uvedené jako první jsou defaultními hodnotami. V příkazové řádce pro použití tohoto software pak stačí zadat tento příkaz v následujícím tvaru:

run_tagger tagger_model inputfile

Konkrétně na Linuxu s nastavením výstupu ve formátu xml do souboru output.xml vygeneruje příkaz „*./run_tagger ../czech-tagger/czech-morfflex.tagger input.txt --output=xml > output.xml*“ následující výstup:

```
<sentence>
    <token lemma="pokladní" tag="NNMS1-----A----">Pokladní</token>
    <token lemma="já" tag="PH-S3--1-------">mi</token>
    <token lemma="odmítnout_:W" tag="VpYS---XR-AA--1">odmítl</token>
    <token lemma="víno" tag="NNNS4-----A----">víno</token>
    <token lemma="vydat-
4_:W_^(platit,_př._vydat_peníze,_v._se_ze_všeho)" tag="Vf--------A--
--">vydat</token>
```

Podobně se vygeneruje vertikální výstup při změně parametru output na vertical. Výstupem jsou tři sloupečky oddělené tabulátorem, tedy původní slovo, lemma a vygenerovaný tagset. Výstupní formát *csv*[5] je vhodný pro další zpracování např. softwarem Microsoft Excel či jiným tabulkovým editorem.

```
Pokladní   pokladní        NNMS1-----A----
mi         já              PH-S3--1-------
```

[5] Formát csv (Comma-separated values, tedy čárkami oddělené hodnoty). Často se k oddělení hodnot používá též středník nebo tabulátor

```
odmítl          odmítnout_:W    VpYS---XR-AA--1
víno       víno             NNNS4-----A----
vydat      vydat-4_:W_^(platit,_př._vydat_peníze,_v._se_ze_všeho)
     Vf--------A----
```

Morphodita nabízí možnost konverze do různých tagsetů a guesser (tj. boolean, hodnota zda má software odhadovat neznámé lemma, popř. tagy, či nikoliv) viz. nápověda softwaru.

Taktéž je možné Morphoditu použít přímo do svého projektu jako knihovnu. K dispozici je rozsáhlá dokumentace.

3.6.5. Lemming a Marmot

Dokumentace uvádí následující způsob vedoucí ke spuštění programu:

```
java -Xmx5g -cp marmot.jar:trove.jar marmot.morph.cmd.Annotator -
model-file ./cs.marmot -test-file form-index=0,input.txt -pred-
file ./out.tsv
```

Nápověda k softwaru se objeví zadáním příkazu zvýrazněným tučně výše. Zobrazí se rozsáhlý seznam nápovědy, obsahující šedesát možností modifikací spuštění, z toho jsou povinné pouze tři, a to **model-file** (datový model), **test-file** (vstupní soubor) a **pred-file** (výstupní soubor). Výstupem příkazu s obsahem souboru input.txt „*Sním je místo něho*.“ je následující (Konce příliš dlouhých řádků byly nahrazeny třemi tečkami):

```
1 Sním    _ _ _ N _ N0N|N_N|N|N1c|N_c|c|N2n|N_n|n|N3s|N_s|s|N4i|N_i|i
2 je      _ _ _ V _ V0V|V_V|V|V1c|V_c|c|V2i|V_i|i|V3p|V_p|p|V43|V_3|3|…
3 místo   _ _ _ S _ S0S|S_S|S|S1p|S_p|p|S2s|S_s|s|S3g|S_g|g
4 něho    _ _ _ P _ P0P|P_P|P|P1p|P_p|p|P23|P_3|3|P3m|P_m|m|P4s|P_s|s|P5g|…
5 .       _ _ _ c _ c0c|c_c|c
```

Ani po větší snaze autor nepřišel na možnost, jak nechat vypsat společně s tagy i **lemma** daných slov.

3.6.6.Czech "Free" Morphology

Při zadání příkazu „./FMAnalyze.pl“ do příkazové řádky na Linuxu nad složkou s instalací Czech Morphology se do konzole vypíše návod k použití.

```
./FMAnalyze.pl: version 1.1.0 2001/08/22 stdin/stdout
Usage: ./FMAnalyze.pl Dictionary-file [Mode] [Mode] < In > Out
(or:   ./FMAnalyze.pl NOSTDIO Filein Fileout Dictionary-file [Mode]
[Mode])
```

```
Filtering Modes:
                        news      suitable for news
                        coll      colloquial style exp.
                        old       old Czech
                        spell     heavy filtering
                        all       no filtering (default)
Demo mode:
                        demo      for interactive testing
```

Autor využil použití uvedené v závorce, tj.

```
./FMAnalyze.pl NOSTDIO ./input.txt output.csts ./CZE-a.il2
```

Předchozí příkaz přečte soubor *input.txt* v nezpracované podobě a vytvoří definovaný soubor se zpracovaným výstupem. Autor upozorňuje na důležitost správného kódování znaků. Je třeba, dle dokumentace nástroje, použít vstupní a výstupní kódování jako ISO Latin 2 (ISO 8859-2) nebo Windows 1250 8-bit kódování.

Pro názornou ukázku fungování tohoto software byla vložena do analyzátoru věta „*Sním je místo něho.*“ s následujícím výstupem:

```
<csts>
<f cap>Sním<MMl>sníst<MMt>VB-S---1P-AA---<MMl>snít<MMt>VB-S---1P-AA-
--
<f>je<MMl>být<MMt>VB-S---3P-AA---<MMl>on-1_^(oni/ono)<MMt>PPNS4--3--
-----<MMt>PPXP4--3-------
<f>místo<MMl>místo-1_^(fyzické_umístění)<MMt>NNNS1-----A----
<MMt>NNNS4-----A----<MMt>NNNS5-----A----<MMl>místo-
2_^(záměnou_za)<MMt>RR--2----------
<f>něho<MMl>něha<MMt>NNFS5-----A----<MMl>on-1_^(on)<MMt>P5ZS2--3----
--1<MMt>P5ZS4--3------1
<D>
<d>.<MMl>.<MMt>Z:-------------
</csts>
```

Výstupem je datový soubor ve formátu csts, ve kterém uživatel najde roztokenizovaný text a ke každému slovu jednu nebo více možných interpretací (lemma a tagy).

Pomocí příkazu v příkazové řádce nad složkou s instalací softwaru „./FMAnalyze.pl ./CZE-a.il2 demo“ lze Free Morphology spustit v tzv. interaktivním režimu. V tomto režimu může uživatel psát text do konzole (pozn. autora: je nutné nastavit kódování konzole na akceptovatelné) a po stisknutí klávesy enteru se do konzole vypíší roztokenizovaná slova s jednou nebo více interpretacemi.

Pro příklad vstup „*Mám rád víno*“ vytvoří následující výstup:

```
Mám
      máma
            NNFP2-----A----
      mít
            VB-S---1P-AA---
rád
      rád
            ACYS------A----
víno
      víno
            NNNS1-----A----
            NNNS4-----A----
            NNNS5-----A----
```

4. Výsledky

Následující tabulka poskytuje přehled dostupných softwarů, které provádějí nejen lemmatizaci českého jazyka.

	Tokenizátor	Analyzátor[6]	**Lemmatizátor**	Tagger	Platforma	GUI	Příkazová řádka	Knihovna	Web rozhraní	Licence	Modulární	Podpora a vývoj	Jazyků
Ajka	✗	?	?	✓	C	✗	✓	✗	✗	NLP	✗	✗	1
Majka	✗	✓	✗	✓	Linux Win	✗	✓	✗	✗	GPL2	✗	✗	15
Morče	✗	✓	✗	✓	Linux Win	✗	✓	✗	✗	GPL2	✗	✗	?
LemmaGen	✓	✗	✓	✗	All	✗	✓	✓	✗	OS	✓	✓	15
MorphoDiTa	✓	✗	✓	✓	All	✗	✓	✓	✓	MPL2	✓	✓	2
Lemming Marmot	✗	✗	✗	✓	All	?	✓	✓	✗	?	✗	✓	20
Czech HMM Tagger	✗	?	?	✓	Linux	✗	✓	?	✗	?	✗	✗	1
Czech Morphology	✓	✓	✗	✓	Linux Win	✗	✓	✗	✗	?	✗	✗	1
Morfo	✗	✓	✗	✗	C	✓	✓	✗	✗	GNU	✗	✗	1
Compost Czech	✗	✗	✗	✓	Linux	✗	✓	✗	✗	?	✗	✗	5
RDRPOSTagger	✓	✗	✗	✓	Win Linux	✗	✓	✓	✗	Freeware	✗	✓	40

Tabulka 5: Tabulka lemmatizátorů s uvedenou funkcionalitou [vlastní zpracování]

[6] Mezi analyzátorem a lemmatizátorem je stanoven rozdíl, že analyzátor má mnoho výstupů, tj. nejednoznačná lemmatizace, kdežto výstupem lemmatizátoru je konkrétní lemma daného vstupního slova.

4.1.Výběr kandidátů

Následující lemmatizátory byly z výběrů kandidátů vyloučeni s daným odůvodněním:

Majka i **Czech Morphology** byly vyloučeny, protože na výstupu není jednoznačné lemma, ale všechny možné významy daného slova.

Lemming společně s **MarMoTem** byly vyloučeny pro nemožnost vypsání lemma k danému slovu.

Czech HMM Tagger, protože se ho nepodařilo zprovoznit.

A byl také vyloučen z výběru **Morče**, protože nebyl nalezen žádný nástroj, který zpracuje text do formátu csts na jeho vstup.

Po stručné, povrchní selekci zůstaly ve výběru dva softwary, tj. **MorphoDiTa** a **Lemmagen**.

4.2. Srovnání rychlosti lemmatizátorů

Vybranými lemmatizátory bylo vyzkoušeno zlemmatizování pět definovaných volně dostupných děl. V případě nástroje MorphoDiTa nebylo potřeba zapínat měření času, protože po skončení procesu délku trvání vypíše. LemmaGen čas nevypisuje a bylo potřeba použít Linuxové funkce *time*.

	Povídky malostranské – V Košili	**Povídky z jedné kapsy**	**Staré pověsti české**	**Babička**	**Bible**
Nezpracovaný text					
LemmaGen	0,006s	0,039	0,028s	0,035s	0,143s
MorphoDiTa	0,071s	3,261s	4,959s	4,374s	38,299s
Zpracovaný text (slovo na řádek, věty oddělené prázdným řádkem)					
LemmaGen	0,021s	0,051s	0,048s	0,051s	0,215s
MorphoDiTa	0,064s	3,816s	2,812s	3,544s	26,141s

Tabulka 6: Měření rychlostí lemmatizátorů LemmaGen a MorphoDiTa [vlastní zpracování]

LemmaGen je oproti softwaru MorphoDiTa řádově rychlejší. Nelogický je fakt, že LemmaGen zpracuje nezpracovaný text rychleji, než ten zpracovaný.

Pro zjištění, kolikrát je LemmaGen podle předchozí tabulky průměrně rychlejší, byl použit následující vzorec, který vezme každý z deseti měření a poměří čas softwaru MorphoDiTa a softwaru LemmaGen.

$$k = \frac{\sum_1^{10} \frac{MorphoDiTa}{LemmaGen}}{10} \doteq 99$$

Na provedeném měření dosahuje LemmaGen oproti softwaru MorphoDiTa až stokrát větší rychlosti.

4.3. Expert Choice

Následuje bezpochyby silně subjektivní část. Při práci s nástrojem Expert Choice se uplatňují autorovy znalosti podložené bakalářskou prací a pocity, které nabyl při zkoušení daných lemmatizátorů.

4.3.1. Platforma

Platforma byla kritériem, které lze chápat tak, že čím více platforem je podporovaných, tím je nástroj v ohledu platformy lepší.

	MorphoDiTa	Lemmagen
MorphoDiTa		2,0
Lemmagen	Incon: 0,00	

Obrázek 6: Kritérium platforma, nástroj Expert Choice [vlastní zpracování]

MorphoDiTa dostala více plusových bodů z důvodu, že je schopna, na rozdíl od LemmaGen, běžet jako webový server a přijímat požadavky na lemmatizaci z internetu.

4.3.2. Podpora

	MorphoDiTa	Lemmagen
MorphoDiTa		2,0
Lemmagen	Incon: 0,00	

Obrázek 7: Kritérium podpora, nástroj Expert Choice [vlastní zpracování]

V hodnocení podpory se uplatnil fakt, že MorphoDiTa je v rukou akademické obce ÚFALu a jsou na ní čerpány dotace. O LemmaGenu fakta s dotacemi nejsou známa.

4.3.3. Práce s kontextem

	MorphoDiTa	Lemmagen
MorphoDiTa		7,0
Lemmagen	Incon: 0,00	

Obrázek 8: Kritérium práce s kontextem, nástroj Expert Choice [vlastní zpracování]

V kritériu práce s kontextem vyhrává bez pochybností MorphoDiTa, která je schopná rozpoznat význam slova na základě kontextu věty. Ve výstupu taktéž odlišuje jednu větu od druhé. LemmaGen funguje jako bezkontextový lemmatizátor založený na RDR.

4.3.4. Rozsah textu a rychlost zpracování

	MorphoDiTa	Lemmagen
MorphoDiTa		3,0
Lemmagen	Incon: 0,00	

Obrázek 9: Kritérium kvantita, nástroj Expert Choice [vlastní zpracování]

V kritériu kvantity vítězí LemmaGen, který je podle oficiálních stránek projektu schopný zvládat lemmatizaci s rychlostí až jeden milion slov za vteřinu, což mj. potvrdila autorova zkušenost s LemmaGen při lemmatizování Bible.

4.3.5. Snadnost použití

	MorphoDiTa	Lemmagen
MorphoDiTa		3,0
Lemmagen	Incon: 0,00	

Obrázek 10: Kritérium snadnost použití, nástroj Expert Choice [vlastní zpracování]

V doméně snadnosti použití vítězí MorphoDiTa. Jedná se o český nástroj a má k dispozici rozsáhlou dokumentaci. LemmaGen propracovanou dokumentací nedisponuje a uživatel se musí spolehnout na nápovědu, která je poskytnuta při spuštění programu.

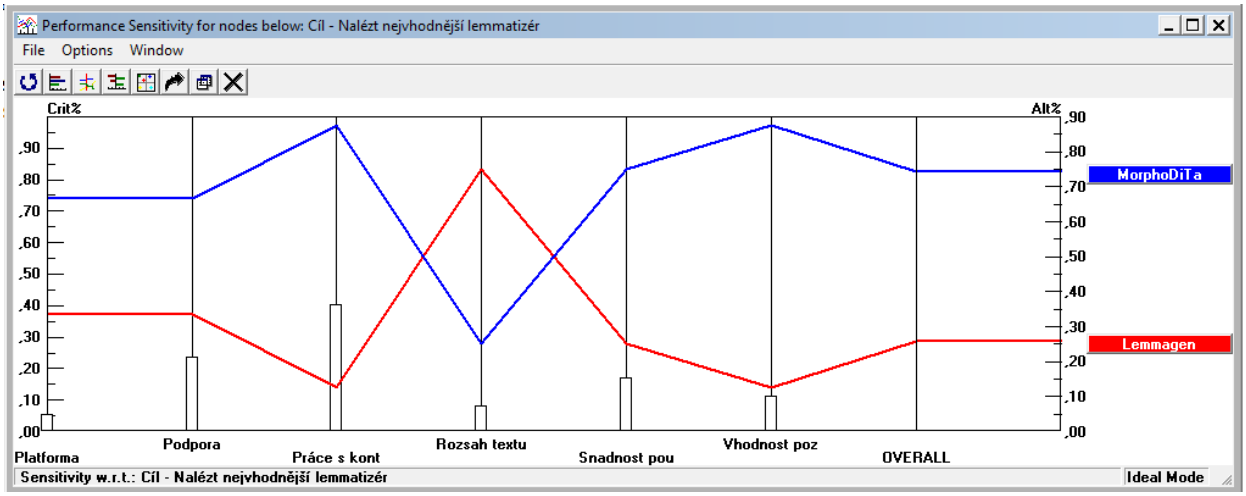
4.3.6. Vhodnost pozičních tagů

	MorphoDiTa	Lemmagen
MorphoDiTa		7,0
Lemmagen	Incon: 0,00	

Obrázek 11: Kritérium vhodnost pozičních tagů, nástroj Expert Choice [vlastní zpracování]

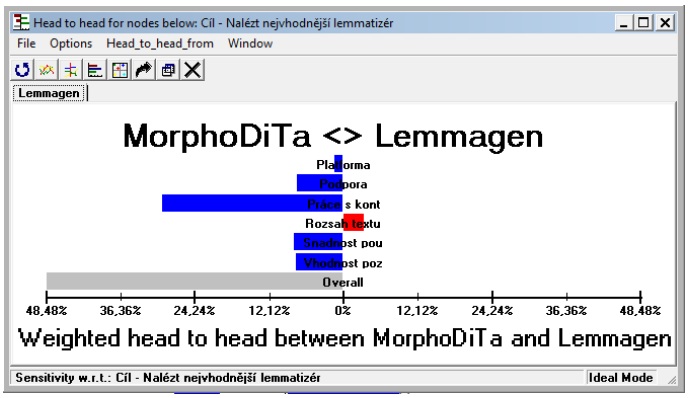
Vzhledem k tomu, že LemmaGen žádné POS tagy neprodukuje, je jednoznačné, že ve vhodnosti pozičních tagů vítězí MorphoDiTa.

4.3.7. Výsledný graf



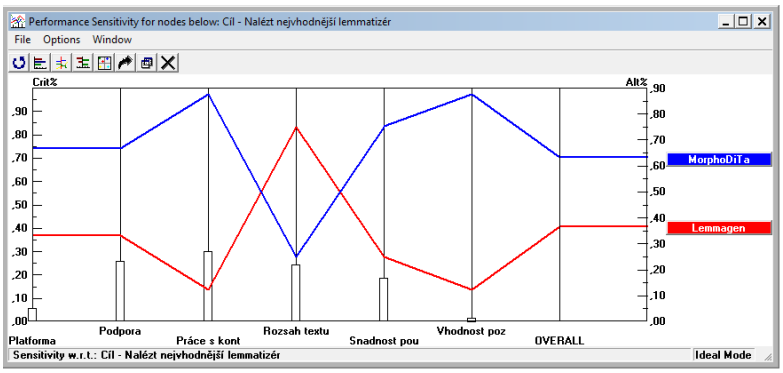
Obrázek 12: Performance sensitivity graph Expert Choice, [vlastní zpracování]

Obrázek č. 12 je výstupem softwaru Expert Choice. Velikost sloupečků jednotlivých kritérií znázorňuje hodnoty vah definovaných v kapitole 3.3. Vzdálenost **modré** a **červené** křivky určuje předchozí porovnání lemmatizátorů v jednotlivých kritériích, viz. kapitoly 4.3.1 – 4.3.6. Na pravé straně grafu je výsledné pořadí lemmatizátorů.



Obrázek 13: Porovnání lemmatizátorů v kritériích, Expert Choice [vlastní zpracování]

Porovnání lemmatizátorů MorphoDiTa a Lemmagen je vidět i na obrázku č. 13, z kterého je patrné, že přidává využitelnosti na stranu softwaru MorphoDiTa.



Obrázek 14: Citlivostní analýza - změna vah, Expert Choice [vlastní zpracování]

Sensitivity performance graph v nástroji Expert Choice umožňuje v poslední fázi rozhodování měnit váhy jednotlivých kritérií, podobně jak bylo učiněno autorem na obrázku č. 14. Autor se rozhodl snížit významnost kritérií ohledně vhodnosti pozičních tagů a vyzdvihnout význam rozsahu a rychlosti zpracování textu.

Na pravé straně se k sobě MorphoDiTa a LemmaGen přiblížily. Rozhodnutí, jestli použít nástroj jeden nebo druhý, se pro rozložení vah kritérií, jak znázorňuje obrázek č. 14, stává obtížnějším.

5. Diskuze

Výsledkem bakalářské práce je sestavené pořadí vhodných lemmatizátorů pro definované požadavky. Jako první se umístila MorphoDiTa, která na autora působila naprosto spořádaným a profesionálním dojmem, druhým v pořadí se stal software LemmaGen, který naopak vyniká rychlostí zpracování textu, je až stokrát rychlejší než MorphoDiTa.

Cíl práce, nalezení nejvhodnějšího nástroje, podle definovaných požadavků, které byly orientovány na kvalitu zpracování, byl splněn.

LemmaGen se díky své rychlosti a jednoduchosti v použití může hodit do projektů, kde nezáleží na kvalitě lemmatizace, ale na rychlosti zpracování. Pokud opravdu zvládá milion slov za vteřinu, jak říkají oficiální stránky projektu, což mj. potvrzují výsledky měření, pak autor nepochybuje o použití lemmatizátoru ve full textových vyhledávačích nebo např. v indexovacím softwaru.

Co se týče samostatného teoretického textu práce, má autor několik **poznámek** a **návrhů.**

Hned zpočátku práce **nesouhlasí** s tím, že je stemmatizace pro angličtinu dostatečná. V angličtině se vyskytují výrazy, jako *to be (am, is, are)* nebo *good (good, better, best),* u kterých se převedením slova na kořen nedosáhne výsledku základního tvaru slova. Nebo např. slovo *break*, které je mnohoznačné (ve významu *pauza, zlomit, útěk z vězení, break dance, break down,* atd.) a pro jeho jednoznačnou interpretaci potřebujeme více, než jen stemming.

Tvrzení, že založení lemmatizátoru pouze na algoritmech hrubé síly (rozumíme použití pouze slovníků) je výpočetně i paměťově náročné, **vyvrací** použití lemmatizátoru pouze na základě slovníku a **potvrzuje** použití hybridních lemmatizátorů, které spojují výhody strojového učení Ripple Down Rules a slovníky pro specifické případy jako vlastní jména nebo zkratky. Jako rozumná by se mohla jevit implementace slovníku Ispell [8].

V metodě Ripple Down Rules autor **spatřuje výhodu** v nízké výpočetní náročnosti. Při hledání lemmatu totiž stačí projít jedinou specifickou cestou skrz celý strom a zmenšit tak počet operací potřebných k nalezení lemmata na nutné minimum.

V kapitole o tokenizaci se píše o slovech, která jsou z procesu lemmatizace vynechána, konkrétně *prosím, prostě, proti, protože* a tak dále. Podle autorova názoru **by bylo zajímavé** zkoumat závislosti používaných výplňkových slov a různých diagnóz či poruch. Znalost

rozpoznání diagnóz z psaného textu by mohla působit preventivě a podle analyzování textů dostupných na internetu zajistit nebezpečné situace.

Z důvodů **specifik** češtiny, konkrétně homografie, homonymie, idiomů, vlastních jmen, neologismů, anglicismů, nářečí, dialektu, nespisovné psané češtiny či překlepů nebo dalších specifik v českých textech se autor domnívá, že nikdy nebude existovat stoprocentní lemmatizátor. Jako například MORČE [35], které se podle oficiálních webových stránek přibližuje k 96% spolehlivosti.

Vyhledávání lemmatizátoru na internetu **na autora působilo celkovým chaotickým dojmem.** Lingvistické nástroje, zdá se, vznikají dost spontánně a získat někde přehled o všech dostupných nástrojích, popř. jejich porovnání, působilo nemožně.

Lemmatizátory se odlišují nejen kvalitou a kvantitou výstupu, ale i náročností použití, za použití rozdílných technologií, s různými licencemi a ne všechny jsou aktualizované a otevřené dalším úpravám.

Překvapující pro autora je, že i přes vynaloženou snahu nenašel komerční lemmatizátor. Všechny lemmatizátory zmíněné v bakalářské práci jsou pod licencí, která umožňuje použití v nekomerčním prostředí. Pouze datové modely, které jsou nositeli znalostí, se distribuují pod licencí umožňující pouze nekomerční použití.

Autor kladně hodnotí **volné** použití všech softwarů, obzvláště pak snahu vytvářet lemmatizátory jako multiplatformní nástroje.

Naopak zklamáním pro autora byl software Lemming, který přes všechny možnosti nastavení (40) se stává nepoužitelným pro svou složitost.

Co se týče využití lemmatizace v praxi, velice **užitečnou aplikací** lemmatizace se může stát strojové zpracování dotazníků s otevřenými otázkami a zobrazování odpovědí do grafů.

Nemalý potenciál autor spatřuje ve spojení sémantického webu a lemmatizace. S rozvojem ontologií a sémantického webu autor spatřuje tendence rozumění stroje lidskému jazyku. Tokenizace, lemmatizace, ale i určování pozičních tagů bezpochyby pomůže výpočetní technice porozumět slovům podle kontextu, v jakém se vyskytují.

Hypotetickým využitím procesu lemmatizace může podle autora být nástroj, který bude hlídat významy slov a vět a upozorňovat na případná rozporná tvrzení v jednom dokumentu. Aplikaci využijí právníci, popř. zákonodárci při tvorbě zákonů.

6. Závěr

Bakalářská práce tvoří průvodce problematiky lemmatizování češtiny, konkrétně co se strojového zpracování týče. Vysvětluje základní i pokročilé pojmy z oboru lingvistiky a seznamuje čtenáře s principem fungování lemmatizátorů, od tokenizace až po přidělování pozičních tagů. Uvádí techniku algoritmů hrubé síly nebo Ripple Down Rules.

Mezi vybrané lingvistické nástroje patří následující softwary: Ajka, Majka, MORČE, LemmaGen, MorphoDiTa, Lemming a MarMoT, Czech HMM Tagger, Czech Free Morphology, QUITA, Morfo, Compost Czech, RDRPOSTagger, Ant, MorphCon a Interset.

Pro vytvoření pořadí mezi selektovanými nástroji byl použit software pro podporu manažerského rozhodování **Expert Choice**.

Jeden z cílů, vyzkoušet a popsat lingvistické nástroje, tj. všechny výše jmenované, byl splněn.

Autor stanovil šest kritérií, a to platformu, podporu vývoje, práci s kontextem (kvalita), rozsah a rychlost zpracovávaného textu (kvantita), snadnost použití a vhodnost pozičních tagů. Nejdůležitějším se stalo kritérium kvality zpracování.

Vzhledem k tomu, že většina vyjmenovaných nástrojů neměla na výstupu jedno jednoznačné lemma, k finálnímu zhodnocení lemmatizátorů se dostaly pouze **LemmaGen** a **MorphoDiTa**, u kterých byla zhodnocena rychlost zpracování textů. Oba dva nástroje splňují akceptační kritérium volného použití a podporovaný vývoj.

Hlavní cíl práce, a to nalézt nejvhodnější nástroj podle zadaných kritérií, byl splněn. Nástroje byly porovnány mezi sebou a vítězem na základě definovaných kritérií se stal nástroj ÚFALu **MorphoDiTa**, který nabízí široké možnosti svého využití.

.

7. Seznam obrázků a tabulek

8. Seznam zdrojů

[1] ANTHONY, Laurence. Software. Laurence Anthony's Websites [online]. [cit. 2016-12-25]. Dostupné z: http://www.laurenceanthony.net/software.html

[2] Český akademický korpus 2.0. Průvodce českým akademickým korpusem 2.0 [online]. [cit. 2016-11-29]. Dostupné z: https://ufal.mff.cuni.cz/rest/CAC/doc-cac20/cac-guide/cz/html/ch3.html

[3] CVRČEK, Václav a RICHTEROVÁ, Olga. Pojmy:{lemma, tag, desambiguace, struktura korpusu, morfologická analýza}. In: Wiki: Český národní korpus [online]. Příručka ČNK, 2013-15 [cit. 2016-06-21]. Dostupné z: http://wiki.korpus.cz/doku.php/pojmy:{lemma, tag, desambiguace, struktura_korpusu, morfologicka_analyza}

[4] Český národní korpus: Kdo jsme? [online]. [cit. 2016-06-21]. Dostupné z: https://www.korpus.cz/

[5] DIBLÍK, Ondřej a KUKUČOVÁ, Simona. 2012. „Homonymie.“ Encyklopedie lingvistiky, ed. Kateřina Prokopová [online]. Olomouc: Univerzita Palackého v Olomouci. [cit. 2016-06-21]. Dostupné z: http://oltk.upol.cz/encyklopedie/index.php5/Homonymie

[6] HAJIČ, Jan. Czech "Free" Morphology: Jan Hajič, 2000-2001 [online]. [cit. 2017-01-03]. Dostupné z: http://ufal.mff.cuni.cz/pdt1/Morphology_and_Tagging/Morphology/index.html

[7] Interset: Interlingua for Morphosyntactic Tagsets. Institute of Formal and Applied Linguistics: Charles University, Czech Republic Faculty of Mathematics and Physics [online]. [cit. 2017-02-05]. Dostupné z: https://ufal.mff.cuni.cz/interset

[8] Ispell [online]. [cit. 2016-06-21]. Dostupné z: https://github.com/tvondra/ispell_czech

[9] Jazykové nástroje Lingea. Překlady.cz [online]. [cit. 2016-06-21]. Dostupné z: http://www.preklady.cz/jazykove-nastroje

[10] JURŠIČ, M.; LemmaGen – Multilingual Open Source Lemmatization. Jožef Stefan Institute, Department of knowledge technologies. 2010. [online] Dostupné z www: http://lemmatise.ijs.si/Services

[11] JURŠIČ, Matjaž, Igor MOZETIČ, Tomaž ERJAVEC a Nada LAVRAČ. Journal of Universal Computer Science: LemmaGen: Multilingual Lemmatisation with Induced Ripple-

Down Rules [online]. 2010, 9(16) [cit. 2016-06-22]. Dostupné z: http://kt.ijs.si/nada_lavrac/Publications/JUCS-2010-LemmaGen.pdf

[12] KARÁSEK, Jan, Ondřej MORSKÝ, Pavel ŠANDA a Radim BURGET. Strojové učení základem pro hybridní lemmatizační algoritmus. Elektrorevue [online]. 2012, (5), 1-11 [cit. 2016-06-21]. ISSN 1213-1539. Dostupné z: http://www.elektrorevue.cz/cz/download/strojove-uceni-zakladem-pro-hybridni-lemmatizacni-algoritmus/

[13] KARLÍK, P.; NEKULA, M.; RUSÍNOVÁ, Z. (eds.). Příruční mluvnice češtiny. Praha: Nakladatelství Lidové noviny, 1995. ISBN 80-7106-134-4.

[14] KOSEK, Jiří. SGML: Standard Generalized Markup Language [online]. 1999 [cit. 2017-02-14]. Dostupné z: http://www.kosek.cz/clanky/cw/sgml.html

[15] KRBEC, Pavel. Czech HMM-based Tagger (using full morphology): Pavel Krbec (HMM Tagger) 2001, Jan Hajic (morphology) 2001 [online]. [cit. 2017-01-01]. Dostupné z: http://ufal.mff.cuni.cz/pdt1/Morphology_and_Tagging/Tagging/MM_tagger/index.html

[16] LemmaGen: Multilingual Open Source Lemmatisation [online]. [cit. 2016-11-29]. Dostupné z: http://lemmatise.ijs.si/

[17] Lemming - A flexible and accurate lemmatizer [online]. Thomas Müller, 2015 [cit. 2017-02-07]. Dostupné z: http://cistern.cis.lmu.de/lemming/

[18] MarMoT - A fast and accurate morphological tagger [online]. Thomas Müller, 2013 [cit. 2017-02-10]. Dostupné z: http://cistern.cis.lmu.de/marmot/

[19] MATERNA, Jiří. Sémantická analýza textů [online]. 2011, ({3,4}) [cit. 2016-06-21]. Dostupné z: http://vyhledavani.sblog.cz/{2011/09/22/semanticka-analyza-textu-3/, 2011/10/17/semanticka-analyza-textu-4/}

[20] MATLACH, Vladimír; Kvantitativně lingvistický software. Diplomová práce, Olomouc. Univerzita Palackého v Olomouci, Fakulta Filozofická.

[21] Morče - Czech Morphological Tagger. Institute of Formal and Applied Linguistics: Charles University, Czech Republic Faculty of Mathematics and Physics [online]. [cit. 2016-11-29]. Dostupné z: http://ufal.mff.cuni.cz/morce/index.php

[22] Morfo – Czech Morphological Analyzer: A project of the Institute of Formal and Applied Linguistics, Faculty of Mathematics and Physics, Charles University [online]. [cit. 2017-01-27]. Dostupné z: http://ufal.mff.cuni.cz/morfo/#about

[23] Morfologické značkování. Institute of Formal and Applied Linguistics Wiki [online]. [cit. 2017-04-05]. Dostupné z: https://wiki.ufal.ms.mff.cuni.cz/user:zeman:ukoly:hmm-tagger

[24] MorphoDiTa. Institute of Formal and Applied Linguistics: Charles University, Czech Republic Faculty of Mathematics and Physics [online]. [cit. 2017-03-10]. Dostupné z: http://ufal.mff.cuni.cz/morphodita

[25] NGUYEN D. Q. RDRPOSTagger: A Rule-based Part-of-Speech and Morphological Tagging Toolkit [online]. Dat Quoc Nguyen, Dai Quoc Nguyen, Dang Duc Pham, and Son Bao Pham [cit. 2017-04-14]. Dostupné z: http://rdrpostagger.sourceforge.net/

[26] NGUYEN, D.Q., NGUYEN, D.Q., PHAM, D.D. and PHAM, S.B. (2016). "A Robust Transformation-Based Learning Approach Using Ripple Down Rules for Part-Of-Speech Tagging." AI Communications, vol. 29, no. 3, pages 409-422. [.pdf]

[27] OSOLSOBĚ, Klára. Vydala Masarykova univerzita roku 2011. Tisk: Reprocentrum,a.s., vydání 1., 2011, ISBN: 978-80-210-5565-0, ISSN: 1211-3034

[28] PETKEVIČ, Vladimír. Časopis pro moderní filologii: Slovnědruhová a morfologická homonymie, homofonie a homografie v současné češtině [online]. 2015, (2) [cit. 2016-06-21]. ISSN 2336-6591. Dostupné z: http://cmf.ff.cuni.cz/sites/default/files/Vladimir%20Petkevic_127-135.pdf

[29] POŘÍZKA, Petr. MorphCon. Corpus Linguistics [online]. Pořízka Petr, Schäfer Marek, Zeman Daniel, 2008 [cit. 2017-01-03]. Dostupné z: http://corpus.upol.cz/morphcon

[30] SKOUMALOVÁ, Hana. Desambiguátor [online]. Dostupné [2016-06-21] z: http://utkl.ff.cuni.cz/desamb-1/

[31] SKOUMALOVÁ, Hana. Positional morphological tags. Ústav teoretické a komputační lingvistiky: Filozofická fakulta, Univerzita Karlova v Praze [online]. [cit. 2016-06-21]. Dostupné z: http://utkl.ff.cuni.cz/~skoumal/morfo/

[32] Projekt (obecné informace o projektu): 1ET101120503. VAVAI.tacr.cz [online]. [cit. 2016-11-29]. Dostupné z: https://vavai.tacr.cz/isvav/project/41995/

[33] PROKOPOVÁ, Kateřina. Manuál Encyklopedie lingvistiky. Olomouc: Univerzita Palackého v Olomouci, 2013. 245 s. ISBN 978-80-244-4129-0. (česky)

[34] RAAB, Jan. COMPOST Czech: Institute of Formal and Applied Linguistics Faculty of Mathematics and Physics, Charles University, Prague, Czech Republic. Institute of Formal and Applied Linguistics [online]. [cit. 2017-01-28]. Dostupné z: http://ufal.mff.cuni.cz/compost/cz/

[35] RAAB, Jan: Morče - Czech morphological tagger. Software prototype, ÚFAL MFF UK, Prague, Czech Rep., http://ufal.mff.cuni.cz/morce, Dec 2007

[36] ROBERTS, Andrew: jTokenizer [online]. [cit. 2016-10-11]. Dostupné z: http://www.andy-roberts.net/coding/jtokenizer/

[37] SEDLÁČEK, R.; Morfologický analyzátor češtiny. Diplomová práce, Brno. Masarykova Univerzita, Fakulta Informatiky.

[38] STRAKA, Milan, STRAKOVÁ, Jana, 2014, MorphoDiTa: Morphological Dictionary and Tagger, LINDAT/CLARIN digital library at the Institute of Formal and Applied Linguistics, Charles University, http://hdl.handle.net/11858/00-097C-0000-0023-43CD-0

[39] STROSSA, Petr. Český lemmatizátor Proč a hlavně jak? [online]. [cit. 2016-06-21]. Dostupné z: http://computerworld.cz/archiv/cesky-lemmatizator-proc-a-hlavne-jak-18850

[40] ŠMERK, Pavel. Fast Morphological Analysis of Czech. In Petr Sojka and Aleš Horák. Proceedings of Third Workshop on Recent Advances in Slavonic Natural Language Processing, RASLAN 2009. Brno: Masaryk University, 2007. p. 13–16. ISBN 978-80-210-5048-8.

[41] The Prague Dependency Treebank 2.0 [online]. [cit. 2017-03-10]. Dostupné z: https://ufal.mff.cuni.cz/pdt2.0/

[42] Volné e-knihy ke stažení zdarma. Volneknihy.xf.cz [online]. [cit. 2017-03-05]. Dostupné z: http://www.volneknihy.xf.cz/

[43] Welcome to Cistern [online]. [cit. 2017-02-02]. Dostupné z: http://cistern.cis.lmu.de/