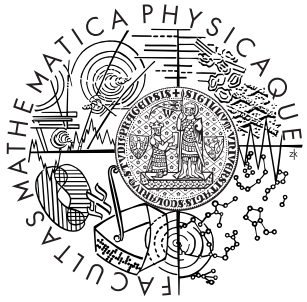
Univerzita Karlova v Praze
Matematicko-fyzikální fakulta

BAKALÁŘSKÁ PRÁCE



Vincent Kríž

Určování syntaktické smysluplnosti českých vět

Ústav formální a aplikované lingvistiky

Vedoucí bakalářské práce: Mgr. Barbora Vidová Hladká, Ph.D.

Studijní program: Informatika, správa počítačových systémů

2009

Chcel by som poďakovať mojej vedúcej Mgr. Barbore Vidovej Hladkej, Ph.D. za skvelý prístup a podporu pri písaní tejto práce.

Prehlasujem, že som svoju bakalársku prácu napísal samostatne a výhradne s použitím citovaných prameňov. Súhlasím s požičiavaním práce a jej zverejňovaním.

V Prahe dňa 20. 4. 2009 Vincent Kríž

Obsah

Zoznam obrázkov

Zoznam tabuliek

Názov práce: Určování syntaktické smysluplnosti českých vět
Autor: Vincent Kríž
Katedra (ústav): Ústav formální a aplikované lingvistiky
Vedúca bakalárskej práce: Mgr. Barbora Vidová Hladká, Ph.D.
E-mail vedúcej: hladka@ufal.mff.cuni.cz

Abstrakt: Určovanie syntaktickej zmysluplnosti viet je zaujímavou a užitočnou úlohou v aplikáciách počítačového spracovania prirodzeného jazyka, napríklad v strojovom preklade, vyhľadávacích strojoch a v systémoch zodpovedania otázok.

Teoretická lingvistika skúma prirodzený jazyk ako systém rovín. V našom projekte tento pohľad rešpektujeme a berieme do úvahy pri definovaní zmysluplnosti. Zmysluplnosť skúmame na základe morfologickej a syntaktickej roviny. V práci implementujeme znalostnú (pravidlovú) procedúru, ktorá o reťazci českých slov rozhodne, či je zmysluplný, alebo nie. Pred spustením procedúry bude reťazec slov analyzovaný externými modulmi, ktoré dodajú morfologické a syntaktické informácie o reťazci.

Cieľovým jazykom je čeština.

Kľúčové slová: zmysluplnosť viet, morfologická rovina, syntaktická rovina

Title: Syntactically-based classification of Czech sentences
Author: Vincent Kríž
Department: Institute of Formal and Applied Linguistics
Supervisor: Mgr. Barbora Vidová Hladká, Ph.D.
Supervisor's e-mail address: hladka@ufal.mff.cuni.cz

Abstract: Classification of syntactically meaningful sentences is a very useful task for the applications of natural language processing, for example machine translation, search engines and question answering systems.

The theoretical linguistic research considers the language to be a system of layers. In our project, a term 'to-be-meaningful' will be specified with respect to this point of view. Namely, the morphological and syntactic layers will be considered. A knowledge-based algorithm classifying a string of Czech words being either meaningful or meaningless will be proposed and implemented. Before being classified, strings will be pre-processed by the external modules.

Czech will be used as the object language.

Keywords: meaningful sentence classification, morphological layer, syntactic layer

Kapitola 1

Úvod

Určovanie zmysluplnosti viet je zaujímavou úlohou v aplikáciách počítačového spracovania prirodzeného jazyka. Napríklad v internetovom vyhľadávaní, alebo v aplikáciách, kde je, okrem iného, potrebné odhaliť zhluk slov určených na zmätenie vyhľadávača.

V práci sa pokúšame nájsť a implementovať procedúru, ktorá v reálnom čase rozpozná, či reťazec českých slov je alebo nie je platnou vetou českého jazyka. Procedúra nebude implementovať žiadny z algoritmov strojového učenia. Jej výsledky nebudú závislé na počte viet, ktoré už doteraz spracovala, ani na množine správnych viet, ktoré by slúžili ako vzory zmysluplných viet.

Procedúra, pri analýze viet českého jazyka, použije výstup externých modulov, ktoré dodajú morfologické a syntaktické informácie. Moduly, ktoré v práci používame, sú šúčasťou Českého akademického korpusu 2.0 a sú obsiahnuté v aplikácii `tool_chain`. Jeho výstupom je súbor vo formáte CSTS. CSTS súbor bude vstupným súborom našej procedúry. Podrobné informácie o Českom akademickom korpuse nájde čitateľ v [1] alebo v [2]. Formát CSTS je podrobne popísaný v [3].

V práci sledujeme systém rovín jazyka podľa konceptu dodržiavaného v Pražskom závislostnom korpuse. Naším cieľom je zistiť, či pri otázke určovania zmysluplnosti je efektívne postupovať po jednotlivých rovinách - od morfologickej, po sémantickú. Vytvorená procedúra s dostatočnou úspešnosťou by bola silným súperom postupu, ktorý sa pri riešení otázky zmysluplnosti ponúka - vytvoriť databázu zmysluplných viet a zadanú vetu porovnať s vetami z tejto databázy. Takáto procedúra je totiž náročná pamäťovo i časovo. V práci sa pokúšame prísť na to, či sledovanie systému rovín nie je príliš *silným kalibrom* na riešenie zmysluplnosti.

Presný popis údajov získavaných z CSTS súborov je popísaný v druhej kapitole tejto práce. Okrem toho, kapitola obsahuje aj podrobnú analýzu zadania a popis problémov, s ktorými sme sa pri riešení stretli. Kapitola sa v neposlednom rade zaoberá otázkou definovania zmyslupnosti vety a stanovením pravidiel na určovanie výsledkov našej procedúry.

Popis predkladaného riešenia nájde čitateľ v kľúčovej tretej kapitole. Rozoberáme v nej jednotlivé algoritmy a pravidlá použité pri riešení zadania.

Štvrtá kapitola obsahuje užívateľskú a programátorskú dokumentáciu k implementovanej procedúre.

V piatej kapitole uvádzame štatistiky implementovanej procedúry, ktoré sme získali analýzou testovacích údajov a zamýšľame sa nad získanými výsledkami.

V šietej kapitole popisujeme webovú implementáciu aplikácie.

Siedma kapitola je zhrnutím práce, kde sú subjektívne popísane jej klady, zmienené niektoré nedostatky a špecifikované ďalšie možné rozšírenia aplikácie.

V dodatku A ponúkame podrobný popis pravidiel použitých v implementovanej procedúre.

Na priloženom CD-ROM môže čitateľ nájsť zdrojové kódy k implementovanej procedúre, vrátane podrobnej programátorskej dokumentácie generovanej systémom Doxygen. Okrem toho, CD-ROM obsahuje aj PDF verziu tejto práce a dokumentu [4]. Popis adresárovej štruktúry CD-ROM je uvedený v dodatku B.

Kapitola 2

Analýza zadania a definovanie zmysluplnosti

V tejto kapitole sa zaoberáme analýzou zadania. Popíšeme vlastnosti, ktoré budeme od procedúry očakávať a na záver vyslovíme definíciu zmysluplnosti tak, ako ju bude chápať implementovaná procedúra.

2.1 Vstup procedúry

Ako sme už povedali v úvode práce, procedúra na vstupe neočakáva reťazec slov, ktoré má analyzovať, ale súbor vo formáte CSTS, ktorý okrem samotnej vety obsahuje aj morfologické a syntaktické informácie týkajúce sa danej vety.

Tieto informácie sú generované externými modulmi, ktoré sú súčasťou Českého akademického korpusu 2.0. Konkrétne sa jedná o nástroj `tool_chain`, ktorý združuje viacero nástrojov do jednej aplikácie. Pre potreby našej aplikácie budeme musieť postupne použiť všetky ponúkané nástroje.

Začneme tzv. *tokenizátorom*, ktorý zadaný text rozdelí na vety a vety na slovné jednotky (*tokeny*). Následne `tool_chain` spustí nástroj na *morfologickú analýzu*. Ten každému z tokenov priradí morfologickú značku, ktorá obsahuje kompletnú morfologickú informáciu o danom vetnom tokene. Keďže tvar tokenu nemusí byť jednoznačný (napríklad slovo *ženu* môže byť jednak podstatné meno a jednak sloveso), morfologická analýza priradí k tokenu všetky možné morfologické značky, ktoré pripadajú do úvahy. Procedúra *tagovania* má potom za úlohu vybrať z pomedzi ponúkaných morfologických značiek tú, ktorá sedí do kontextu tokenu vo vete. Posledným nástrojom, ktorým sa ukončí analýza vety, je procedúra *parsovania*. Jej výstupom je závislostný strom, ktorý vyjadruje syntaktickú informáciu o danej vete. Po tejto analýze sa všetky získané informácie uložia do súboru vo formáte CSTS.

Formát CSTS (z anglického *Czech sentence tree structure*) je vedľajším údajovým formátom Českého akademického korpusu 2.0. Jedná sa o formát SGML používaný v Pražskom závislostnom korpuse a tiež v Českom národnom korpuse. Medzi jeho výhody patrí dobrá čitateľnosť pre človeka, jeho jednoduché spracovanie nástrojmi a tiež fakt, že niektoré nástroje z Českého akademického korpusu 2.0 pracujú výhradne s týmto formátom. Stručný úvod do CSTS je napríklad v [1], podrobná špecifikácia potom v [3].

```
 1 <s id=sample-input.txt-001-p1s1>
 2 <f id=sample-input.txt-001-p1s1W1-Tm>Dráha<l>dráha
 3    <t>NNFS1-----A----<r>1<g>6<A>Sb
 4 <f id=sample-input.txt-001-p1s1W2-Tm>se<l>se
 5    <t>P7-X4----------<r>2<g>4<A>AuxT
 6 <f id=sample-input.txt-001-p1s1W4-Tm>privatizovat<l>privatizovat
 7    <t>Vf--------A----<r>4<g>6<A>Obj
 8 <f id=sample-input.txt-001-p1s1W6-Tm>nestihne<l>stihnout
 9    <t>VB-S---3P-NA---<r>6<g>0<A>Pred
10 <D>
11 <d id=sample-input.txt-001-p1s1W7-Tm>.<l>.
12    <t>Z:-------------<r>7<g>0<A>AuxK
```

Obr. 2.1: Veta *Dráha se privatizovat nestihne.* vo formáte CSTS

Formát CSTS je dobre čitateľný aj pre človeka. Na obázku 2.1 uvádzame výstup procedúry `tool_chain` pre vetu *Dráha se privatizovat nestihne.*

Veta v CSTS súbore začína značkou `<s>`, ktorá obsahuje jednoznačnú identifikáciu vety. Túto identifikáciu bude používať i naša procedúra. Na ďalších riadkoch sú zobrazené značky zodpovedajúce jednotlivým vetným tokenom. Za značkou `<f>` nasleduje slovo tak, ako je uvedené vo vstupnej vete. Značka `<l>` obsahuje lemmu daného slova.

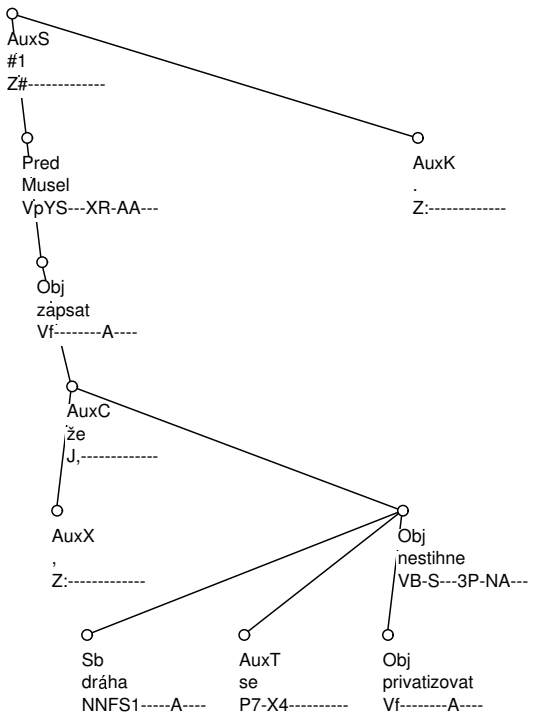
Za značkou `<t>` je umiestnená morfologická značka vo formáte CPMT (z anglického *The Czech Positional Morphological Tags*). V 15-tich znakoch nesie morfologickú informáciu, pričom každý znak zodpovedá jednej gramatickej kategórii. Napríklad, na prvej pozícii je uvedený slovný druh, na druhej slovný poddruh, na tretej rod atď. Podrobná špecifikácia morfologických značiek je súčasťou [3].

Syntaktickú informáciu predstavuje závislostný strom, ktorý je reprezentovaný dvojicou značiek `<r>` a `<g>`. Prvá z nich predstavuje poradie tokenu vo vete, druhá je odkazom na nadradený vetný člen. Pre ľahšiu reprezentáciu stromu, každá veta obsahuje aj technický koreň, ktorý je nadradený slovám, ktoré nie sú nadradené žiadnemu inému slovu vo vete. Na obrázku 2.2 je zobrazený závislostný strom ukážkovej vety *Musel zapsat, že dráha se privatizovat nestihne.* Strom bol zobrazený programom TrEd[1] a je doplnený o morfologické značky a analytické funkcie.

Posledný údaj, ktorý bude mať naša procedúra k dispozícii, je analytická funkcia obsiahnutá v značke `<A>`. Ide o označenie syntaktickej funkcie slova vo vete, ktoré sa v zjednodušenej forme učí už na základných školách. Tabuľku všetkých možných analytických funkcií je možné nájsť napríklad v [1].

Naša procedúra bude používať informácie zo značky `<t>`, ktorá nesie morfologickú informáciu a údaje zo značiek `<r>`, `<g>` a `<A>`, ktoré nesú syntaktickú informáciu.

[1]Program TrEd môže čitateľ nájsť na `http://ufal.mff.cuni.cz/~pajas/tred/`



Obr. 2.2: Závislostný strom vety *Musel zapsat, že dráha se privatizovat nestihne.*

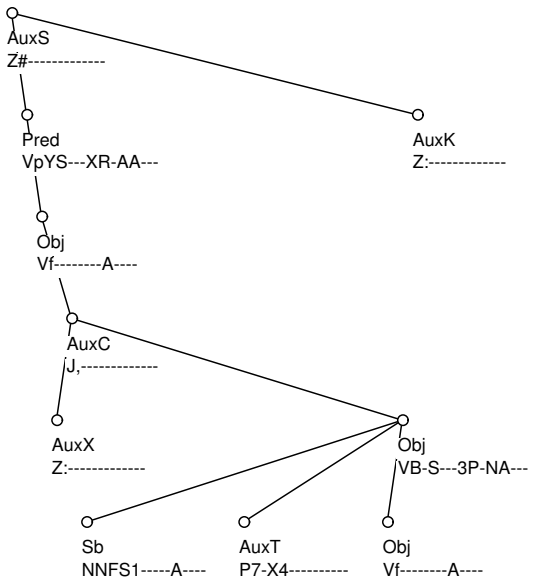
2.2 Roviny jazyka

Teoretická ligvistika, pri skúmaní prirodzeného jazyka, uvažuje o jazyku ako o systéme rovín. Tento pohľad na jazyk sme sa rozhodli zachovať i pri hľadaní odpovede na otázku, ako určovať zmyslupnosť viet. Podľa koncepcie Pražského závislostného korpusu, rozlišujeme 3 roviny jazyka. Každú z nich si priblížime pomocou ukážkovej vety *Musel zapsat, že dráha se privatizovat nestihne.*

Prvou z rovín je *morfologická rovina*. Na tejto úrovni jazyka berieme do úvahy len morfologické informácie o jednotlivých vetných tokenoch. Našu vetu si tak môžeme predstaviť ako reťazec objektov, ktoré sú charakterizované výlučne gra-

```
 1 <s id=sample-input.txt-001-p1s1>
 2 <t>VpYS---XR-AA---
 3 <t>Vf--------A----
 4 <t>Z:-------------
 5 <t>J,-------------
 6 <t>NNFS1-----A----
 7 <t>P7-X4----------
 8 <t>Vf--------A----
 9 <t>VB-S---3P-NA---
10 <t>Z:-------------
```

Obr. 2.3: Morfologická informácia viet *Musel zapsat, že dráha se privatizovat nestihne.* a *Nechtěla vařit, protože rýže se sníst nestihne.* vo formáte CSTS



Obr. 2.4: Syntaktická rovina vety *Musel zapsat, že dráha se privatizovat nestihne.* doplnená o morfologické značky.

matickými kategóriami. Vety s rovnakými morfologickými informáciami by sme nevedeli rozlíšiť. Príkladom nerozlíšiteľných viet na morfologickej rovine môžu byť vety *Musel zapsat, že dráha se letos privatizovat nestihne.* a *Nechtěla vařit, protože rýže se sníst nestihne.* Morfologická informácia vyjadrená vo formáte CSTS, na obrázku 2.3 je rovnaká pre oba vety.

Len pomocou morfologických informácií nerozlíšime nie len niektoré vety, ale ani niektoré slová v rámci vety jednej. V ukážkovej vete sú rovnakou morfologickou značkou označené oba neurčitky slovies (*zapsat* a *privatizovat*), a oba interpunkčné znamienka (čiarka a záverečná bodka). Anotácia na morfologickej rovine týchto dvojíc tokenov je nerozlíšiteľná.

Doplnením morfologickej roviny o vzťahy medzi jednotlivými tokenmi a ich funkcii vo vete sa presúvame do *syntaktickej roviny*. Reprezentácia ukážkovej vety by už mala tvar závislostného stromu. Zobrazená je na obrázku 2.4.

Poslednou rovinou jazyka je rovina *sémantická*. Touto rovinou sa v našej práci nezaoberáme.

V našej práci sledujeme systém týchto jazykových rovín. V ročníkovom projekte, ktorý predchádzal tejto práci sme mali k dispozícii len morfologické informácie. Procedúra, ktorú implementujeme v tejto práci, bude mať k dispozícii aj syntaktické informácie.

2.3 Analýza zadania

Zameranie sa len na prvé dve zo spomínaných troch jazykových rovín nám dáva možnosť odpútať sa od významu slov v jednotlivých vetách. Slová môžme chápať ako objekty, ktoré nenesú význam a sú charakterizované len prostredníctvom ich gramatických kategórií a prostredníctvom vzťahov, ktoré vytvárajú v rámci vety s ostatnými objektmi. V ďalšom texte budeme tieto objekty nazývať *tokeny*.

Morfologické značky

Pri skúmaní morfologických značiek sme zistili, že informácie, ktoré značky nesú na jednotlivých pozíciách, nemusia byť disjunktné. Existujú pozície, na ktorých sa gramatické kategórie vyjadrené značkou môžu prekrývať, alebo, dokonca, môžu byť obsiahnuté vo všeobecnejšej značke.

Ako príklad uvádzame tretiu pozíciu, ktorá vyjadruje rod. Medzi možnými značkami pre tretiu pozíciu nájdeme napríklad `I` pre masculinum inanimatum, `M` pre masculinum animatum, ale aj `Y` pre masculinum, dokonca `X` pre ľubovoľný rod.

Z uvedeného vyplýva, že pri porovnávaní morfologických značiek nebude možné vykonávať prosté porovnanie znakov, ale bude potrebné zisťovať príslušnosť znaku do množiny znakov, ktoré nesú spoločnú vlastnosť.

Úspešnosť procedúry `tool_chain`

Externé nástroje v podobe procedúry `tool_chain`, ktoré používame na získanie vstupu pre našu procedúru, nepracujú so 100% úspešnosťou. Slová, ktoré `tool_chain` nedokáže analyzovať, sú označené morfologickou značkou `X@`. Naša procedúra tak nemusí dostať vždy korektné údaje potrebné pre analýzu zmysluplnosti. Bude preto potrebné vytvoriť postup, ako bude procedúra reagovať na neštandardný vstup.

Okrem chýb, ktorých sa dopustí morfologická analýza a tagger, budeme musieť brať do úvahy aj chyby parsera, ktorý našej procedúre môže dodať nekorektný závislostný strom.

Všetky tieto neštandardné vstupy bude musieť procedúra vhodným spôsobom riešiť.

Formálna kontrola viet

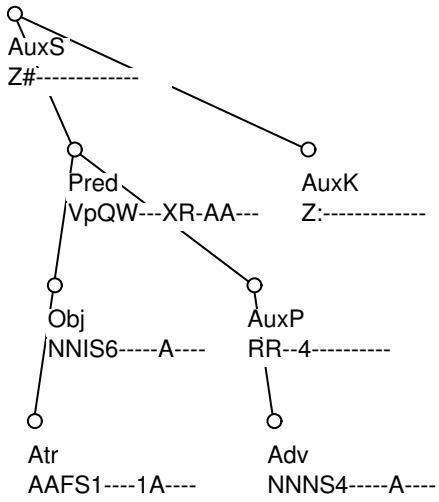
Veta môže mať zmysel aj keď obsahuje gramatické chyby (napríklad, ak sa pred podraďovacou spojkou nevyskytuje čiarka alebo veta nezačína veľkým písmenom). Preto procedúra nebude vykonávať tzv. *formálnu kontrolu viet*.

2.4 Definovanie syntaktickej zmyslupnosti

Zmysluplnosť vety je zväčša chápaná na sémantickej rovine. Veta má zmysel, ak nesie nejakú zmysluplnú informáciu. Keďže naša procedúra nebude pracovať so zmyslom slov, je potrebné definovať zmysluplnosť tak, ako ju chápeme v našej práci.

V sekcii 2.2 sme popísali ako môžeme chápať syntaktickú rovinu vety. Syntakticky zmysluplnú vetu potom môžeme chápať ako *objekt*, kde jednotlivé vetné tokeny ešte nenesú význam, ale len morfologické a syntaktické informácie. Veta bude syntakticky zmysluplná, ak bude existovať aspoň jedna sada slov, ktoré po dosadení do tohto objektu vytvoria (sémanticky) zmysluplnú vetu.

Naša procedúra bude teda analyzovať takého vetné objetky a na základe informácií získaných z CSTS súboru prehlási vetu za zmysluplnú alebo nezmysluplnú.



Obr. 2.5: Objekt vety *Vhodná sněhu nastala pro kolo.*

Príkladom zmysluplnej vety môže byť veta *Nová komedie Marie Poledňákové Líbáš jako Bůh je jasně lepší než její předchozí film Jak se krotí krokodýli.* Za zmysluplnú vetu procedúra však označní napríklad aj *Krátký kabel během letadla zlepší vaši mrkev i výkonnost.* Táto veta nenesie zmysluplný význam, ale zodpovedá našej definícii syntakticky zmyslupnej vety. Za syntakticky nezmysluplnú vetu procedúra označí napríklad vetu *Vhodná sněhu nastala pro kolo.* Do závislostného stromu vety, zobrazenom na obrázku 2.5, totiž neexistuje žiadna sada slov, ktoré by vyhovovali morfologickým a syntaktickým pravidlám a zároveň by niesli sémanticky zmysluplnú informáciu.

Kapitola 3

Popis riešenia

V strojovom spracovaní prirodzených jazykov, napríklad pri kontrole gramatiky, existujú dva základné spôsoby práce, ktorými sa aplikácie riadia. Vo vetách kontrolujú buď správnosť jazykových konštrukcií, alebo, naopak, vyhľadávajú konštrukcie, o ktorých vedia, že sú chybné a v prirodzenom jazyku nemôžu nastať.

Druhý spomínaný spôsob analýzy sme použili aj pri riešení úlohy určovania syntaktickej zmysluplnosti viet. Vo vetách vyhľadávame jakykové konštrukcie, ktoré považujeme za nezmysluplné (v zmysle našej definície zmysluplnosti). Namiesto kontrolovania správnosti vety teda vyhľadávame možné chyby. Tento postup sme zvolili najmä kvôli predpokladu, že chybných konštrukcií, ktoré môžu nastať, je v prirodzenom jazyku menej ako počet všetkých možných správnych konštrukcií.

3.1 Algoritmus procedúry

Procedúra analyzuje každú vetu jednotlivo. V ročníkovom projekte [4], ktorý predchádzal tejto práci, sme riešili otázku, ako analyzovať súvetia. Má zmysel analyzovať každú jednoduchú vetu v súvetí jednotlivo? Nakoniec sme sa rozhodli nedeliť vety v súvetiach. Podobne je tomu aj v tejto práci - súvetia analyzujeme rovnakým spôsobom ako jednoduché vety.

Analýza zmysluplnosti začína kontrolou anotácie na morfologickej rovine. Procedúra zistí všetky chybné konštrukcie na morfologickej úrovni (t.j. len použitím morfologických informácií). Odhalené chybné konštrukcie potom prejdú syntaktickou kontrolou. Zistené chyby sa v poslednom kroku algoritmu spracujú podľa ich závažnosti, na základne ktorej sa veta prehlási za zmysluplnú, alebo nezmysluplnú.

Základom *morfologickej analýzy* je kontrola dvoch (alebo viacerých) po sebe idúcich slov - *kolokácií*. Na základe zoznamu chýb (nepovolených kolokácií) procedúra postupne skontroluje všetky n-tice slov. Morfologická analýza pracuje len s morfologickými informáciami. Znamená to, že kolokácie kontroluje len na základe ich morfologických značiek.

Keďže nepovolených kolokácií je nezanedbateľné množstvo, rozhodli sme sa pravidlá na posudzovanie chybných kolokácií, pre prehľadnosť, rozdeliť do tzv. *morfologických testov*. Pravidlá chybných kolokácií sú rozdelené a testované podľa prvého slovného poddruhu v danej kolokácii. Testy sú následne ešte združené do tzv. *morfologických modulov*, podľa rovnakých slovných druhov. Prehľad morfo-

logických modulov uvádzame v prílohe A, popis metodológie hľadania pravidiel v časti 3.2 tejto práce.

Špeciálnym morfologickým modulom je ModulX, ktorý vo vete vyhľadáva značky `X@----------`. Tieto značky označujú tokeny, ktoré `tool_chain` nedokázal úspešne analyzovať a priradiť im korektnú morfologickú značku.

Chybné kolokácie morfologická analýza ohodnocuje troma úrovňami chýb. Chyba *prvého druhu* je fatálna chyba, ktorej výskyt vo vete považujeme za jasný znak jej nezmysluplnosti. Príkladom testov, ktoré vyhľadávajú tento druh chyby, môže byť napríklad test `A2` v module `A`. Vo vete vyhľadáva prídavné mená typu *technicko*, ktoré `tool_chain` označuje značkou `A2`. Za týmito slovnými poddruhmi musí vždy nasledovať ďalšie prídavné meno. Ak ho test nenájde bezprostredne za značkou `A2`, ide o chybu prvého druhu a veta je určite nezmysluplná.

Chyba *druhého druhu* je chyba, ktorú morfologická analýza zistila na základe zoznamu nepovolených dvojíc slovných poddruhov, ktoré sa vo vete nemôžu vyskytovať za sebou. Chyba druhého druhu je chyba, ktorú vyhľadáva väčšina implementovaných testov. Prehľadávaním Českého národného korpusu (popísaným v časti 3.2) sme získali zoznamy kolokácií, ktoré sa za sebou v českom jazyku nevyskytujú. Príkladom takýchto testov sú napríklad testy modulu `V`.

Na záver, chyba *tretieto druhu* je chyba, pri ktorej slovné poddruhy v kolokácii za sebou nasledovať môžu, nesedia však ich ostatné gramatické kategórie (spravidla rod, číslo a pád). Príkladom testu, ktorý detekuje chyby tretieho druhu je test `NN` v module `N`.

Chybné kolokácie s tretím druhom chyby sú po skončení morfologickej analýzy následne spracované *syntaktickou analýzou*. Tá je tvorená jediným modulom, ktorého úlohou je overiť, či daná kolokácia (spravidla dvojica slov) je v závislostnom strome spojená hranou. Pri riešení tejto otázky analýza využíva informácie zo značiek `<r>` a `<g>` súboru CSTS. Stačí overiť, či prvé slovo v kolokácii je synom alebo otcom druhého slova.

Syntaktická analýza pracuje s dvoma druhmi chýb. Jej vstupom sú kolokácie s chybou tretieho druhu, ktoré overí na existenciu hrany v závislostnom strome. Ak hľadaná hrana neexistuje, syntaktická analýza označí túto kolokáciu chybou *štvrtého druhu*. Ak hrana existuje, chyba kolokácie (tretieho druhu) ostáva nezmenená.

Posledným krokom procedúry je záverečná *analýza chýb*. Spočíva v spočítaní výskytov chybných kolokácií podľa ich druhu a následnom porovnaní získaných výsledkov s nastavenou *citlivosťou*. Tá určuje, koľko chýb, ktorého druhu, je potrebných na prehlásenie vety za nezmysluplnú.

Závislosť výslednej analýzy na nastavenej citlivosti môžeme demonštrovať na vete *Koupil jsem si novou grafickou kartu ATI Radeon Xpress 1100.* Procedúra `tool_chain` označí anglické slová v názve grafickej karty značkami `X@`. Modul `X` následne odhalí oba cudzie slová ako chyby prvého druhu. Ak citlivosť chyby prvého druhu nastavíme na 1, vetu naša procedúra prehlási za nezmysluplnú. Pri citlivosti nastavenej na 3 a viac, bude veta prehlásená za zmysluplnú.

3.2 Metodológia overovania kolokácií

Pravidlá na odhaľovanie chybných morfologických kolokácií sme vytvárali systematicky. Pre každý slovný poddruh sme hľadali pravidlo pre jeho kolokáciu so

všetkými slovnými poddruhmi. Okrem toho sme hľadali aj odpoveď na otázku, či skúmaný slovný poddruh môže byť na konci vety, alebo či za ním môže nasledovať interpunkčné znamienko.

Informácie o možných kolokáciách sme vyhľadávali predovšetkým pomocou nástroja Bonito, ktorým sme prehľadávali Český národný korpus. Výsledky sme schematicky zapisovali do tabuľky kolokácií, ktorá obsahovala políčko pre každú možnú kombináciu dvoch slovných poddruhov.

Výsledky tohto výskumu sú uvedené v podobe morfologických tabuliek v dodatku A.

3.3 Časová a pamäťová zložitosť procedúry

Procedúra spracováva súbor CSTS postupne - vetu po vete. Každú vetu spracováva práve raz. Vetu načíta, analyzuje, a ak je posledná v súbore, končí, inak načíta ďalšiu vetu. Časová zložitosť analýzy CSTS súboru je teda $\mathcal{O}(n)$, kde n je počet viet vo vstupnom súbore.

Analýza vety má, podobne ako analýza celého súboru, zložitosť $\mathcal{O}(n)$ vzhľadom k počtu slov vo vete. Veta sa postupne predáva každému z (konštantného počtu) modulov, ktoré ju predajú (konštantnému počtu) testov. Testy kontrolujú každý vetný token práve raz. Vetu teda každý z testov skontroluje v lineárnom čase.

Celková zložitosť procedúry je teda $\mathcal{O}(cn) = \mathcal{O}(n)$, kde c je počet implementovaných testov a n počet slov vo vstupnom súbore.

Keďže procedúra spracováva v jednom momente práve jednu vetu, vždy má v pamäťových štruktúrach uloženú jedinú, aktuálne spracovávanú vetu. Tú po dokončení analýzy uvoľní z pamäte, aby mohla následne načítať z CSTS súboru ďalšiu vetu určenú na analýzu. Veľkosť pamäte potrebnej na uchovanie vety závisí od počtu tokenov, ktoré veta obsahuje. Pamäťová zložitosť je teda $\mathcal{O}(n)$ vzhľadom na počet slov vo vete.

Kapitola 4

Implementácia - aplikácia SyMorAn

Procedúru popísanú v predchádzajúcich kapitolách sme implementovali v podobe aplikácie SyMorAn. V nasledujúcom texte uvádzame užívateľskú a programátorskú dokumentáciu tejto aplikácie.

4.1 Užívateľská dokumentácia

Aplikácia SyMorAn je textovo orientovaná aplikácia vyvíjaná pre platformu UNIXových operačných systémov. Pracuje štýlom, ktorý je veľmi častý pre *konzolové* aplikácie - pri spustení očakáva definovanie vstupov a ďalších (nepovinných) parametrov, podľa ktorých následne prispôsobí svoj beh. Po spracovaní vstupu skončí. V tabuľke 4.1 uvádzame prehľad všetkých prepínačov, ktorými aplikácia disponuje.

4.1.1 Vstupný súbor

Aplikácia očakáva na vstupe cestu k CSTS súboru určenému na analýzu zmysluplnosti. Cesta sa zadáva ako parameter povinného prepínača `-i`. Cesta sa môže zadať buď v absolútnom alebo relatívnom tvare, vzhľadom na adresár, z ktorého je spúšťaná aplikácia SyMorAn.

4.1.2 Definovanie modulov

SyMorAn umožňuje špecifikovanie morfologických modulov, ktoré sa použijú pri analýze zmysluplnosti vstupných viet. Každý z implementovaných modulov je označený jedinečným znakom. Pre zaradenie modulu do zoznamu modulov, ktoré sa použijú pri analýze zmysluplnosti, stačí znak priradený k modulu pripojiť k parametru prepínača `-m`. V tabuľke 4.2 uvádzame prehľad znakov priradených k jednotlivým morfologickým modulom.

Zoznam všetkých testov a pravidiel, ktoré tieto morfologické moduly obsahujú nájdete v prílohe A tejto práce.

Prepínač	Parameter	Povinný	Význam
`-i`	vstupný súbor	áno	Definuje cestu k vstupnému súboru CSTS
`-m`	popis modulov	nie	Definuje, ktoré morfologické moduly sa pri analýze použijú
`-v`	nemá	nie	Spustí SyMorAn vo verbose móde
`-w`	nemá	nie	Spustí SyMorAn v multiverbose móde
`-l1`	počet chýb	nie	Nastaví citlivosť na chyby 1. druhu
`-l2`	počet chýb	nie	Nastaví citlivosť na chyby 2. druhu
`-l3`	počet chýb	nie	Nastaví citlivosť na chyby 3. druhu
`-l4`	počet chýb	nie	Nastaví citlivosť na chyby 4. druhu
`-h`	nemá	nie	Vytlačí krátku nápovedu

Tabuľka 4.1: Prehľad všetkých prepínačov aplikácie

Znak	Popis modulu
`A`	Kontrola prídavných mien
`C`	Kontrola čísloviek
`J`	Kontrola spojok
`N`	Kontrola podstatných mien
`P`	Kontrola zámen
`V`	Kontrola slovies
`R`	Kontrola predložiek
`X`	Kontrola neklasifikovateľných slov

Tabuľka 4.2: Prehľad morfologických modulov

```
(ID: sample-input.txt-001-p1s1) Musel zapsat , že dráha
se privatizovat nestihne .
OK
(ID: sample-input.txt-001-p1s2) Vhodná sněhu nastala pro kolo .
KO
```

Obr. 4.1: Ukážka výstupu aplikácie v móde *default*.

4.1.3 Definovanie citlivosti aplikácie

Ako sme popísali v časti 3.1, aplikácia detekované chyby rozdeľuje medzi 4 druhy. Citlivosť chýb každého z týchto druhov sa dá nastaviť pomocou prepínačov `-l1`, `-l2`, `-l3` a `-l4` nasledovanými parametrom vyjadrujúcim počet chýb danej úrovne, ktoré musia byť objavené vo vete, aby ju aplikácia prehlásila za nezmysluplnú.

Ak užívateľ nešpecifikuje niektorú z citlivostí, použije sa pre ňu štandardná hodnota 1. Vo východzom stave teda na označenie vety za nezmysluplnú stačí, ak procedúra detekuje aspoň 1 chybu ľubovoľného druhu.

4.1.4 Výstup aplikácie

SyMorAn výstupné údaje zapisuje na štandardný výstup. Na začiatku zobrazí prehľad parametrov, s ktorými bola aplikácia spustená. Nasleduje samotná analýza viet, ktorej tvar sa líši od zvoleného výstupného módu. Na záver aplikácia zobrazí krátku štatistiku analyzovaných viet.

Štatistický prehľad začína počtom viet, ktoré aplikácia analyzovala, nasleduje počet viet (a percentuálny podiel), ktoré prehlásila za zmysluplné a počet viet, ktoré prehlásila za nesmysluplné. Nakoniec je zobrazený počet chýb, ktoré aplikácia detekovala a ich rozdelenie medzi jednotlivými morfologickými modulmi.

Aplikácia umožňuje formátovať výstup analýzy viet v troch módoch. Základným je mód *default*. Tento mód na štandardný výstup vytlačí vetu v čitateľnej podobe (slovo za slovom). Na ďalší riadok potom vytlačí jej hodnotenie. V prípade, že program ohodnotí vetu ako zmysluplnú, vypíše text `OK`, v opačnom prípade text `KO`. Tento mód aplikácia používa ako implicitný a bude použitý, ak užívateľ neurčí inak. Obrázok 4.1 ukazuje výstup programu v móde default pri spracovaní vety *Musel zapsat, že dráha se privatizovat nestihne.* a vety *Vhodná sněhu nastala pro kolo.* Pre úsporu miesta uvádzame len podstatnú časť výstupu aplikácie.

Mód *verbose* vytlačí okrem záverečného hodnotenia zmysluplnosti aj zoznam chýb, ktoré boli pri jednotlivých vetách detekované. Riadok s chybou začína slovom `ERROR`, za ktorým, v zátvorke, nasledujú údaje: úroveň chyby, poradové číslo prvého slova chybnej kolokácie, meno modulu a meno testu, ktorý chybu detekoval. Potom nasleduje stručný popis chyby vrátane morfologických značiek a slov, ktoré sú súčasťou chybnej kolokácie. Príklad tohto výstupu na ukážkové vety *Vhodná sněhu nastala pro kolo.* a *Novinky Galerie obrázků Žebříček obrázků Sbírka vtipů Texty.* je zobrazený na obrázku 4.2. V móde verbose sa aplikácia spustí použitím prepínača `-v`.

Posledným a najpodrobnejším je *multiverbose* mód, v ktorom sa okrem informácií rovnakých ako v móde verbose, zobrazujú aj podrobné informácie o práci jednotlivých modulov. Po zobrazení vety, ktorá sa analyzuje, každý z testov

```
(ID: sample-input.txt-001-p1s1) Vhodná sněhu nastala pro kolo .
ERROR (3, 1, A, AA) Tag AAFS1----1A---- (Vhodná) sa nezhoduje v rode,
cisle a pade s NNIS6-----A---- (sněhu)
KO
(ID: sample-input.txt-001-p1s2) Novinky Galerie obrázků Žebříček
obrázků Sbírka vtipů Texty .
ERROR (4, 3, N, NN) Tag NNIP2-----A---- (obrázků) sa nemoze
vyskytovat pred NNIS1-----A---- (Žebříček)
ERROR (3, 5, N, NN) Tag NNIP2-----A---- (obrázků) sa nemoze
vyskytovat pred NNFS1-----A---- (Sbírka)
ERROR (4, 7, N, NN) Tag NNIP2-----A---- (vtipů) sa nemoze
vyskytovat pred NNIP1-----A---- (Texty)
KO
```

Obr. 4.2: Ukážka výstupu aplikácie v móde *verbose*.

zobrazí, ktorý token kontroloval a s akým výsledkom. Mód sa spustí použitím prepínača `-w`.

4.1.5 Zobrazenie nápovedy

Použitím prepínača `-h` aplikácia zobrazí krátku nápovedu s popisom všetkých prepínačov a so zoznamom morfologických modulov vrátane ich identifikujúcich znakov, potrebných pre prepínač `-m`.

4.2 Programátorská dokumentácia

Aplikácia SyMorAn využíva pokročilé vlastnosti objektového programovania v jazyku C++. V tejto časti práce sa zaoberáme popisom najzaujímavejších objektov a globálnych funkcií, ktoré implementujú procedúru na určovanie syntaktickej zmysluplnosti viet.

4.2.1 Prehľad súborov

Aplikácia SyMorAn je rozdelená do niekoľkých súborov. V nasledujúcom prehľade uvádzame, ktoré súbory implementujú najdôležitejšie objekty a globálne funkcie programu.

- `main.cpp` Obsahuje hlavnú funkciu `main()`, ktorá vykonáva algoritmus procedúry popísaný v časti 3.1, a definície globálnych premenných programu.
- `common.cpp` Obsahuje definície pomocných funkcií, napríklad pre prevod čísel na reťazce a naopak, alebo funkciu `print()`, ktorá sa používa na tlač výstupov programu.
- `config.h` Obsahuje makrá, ktoré nastavujú východzie správanie aplikácie a definície niektorých konštánt programu.

- `moran.cpp` Obsahuje definície všetkých morfologických modulov, vrátane ich spoločného predka a funkcie `zhoda()`, ktorá porovnáva morfologické značky.
- `parsing.cpp` Obsahuje definíciu triedy `Parsing`, ktorá vykonáva parsovanie a preprocessing vstupného CSTS súboru. Tiež obsahuje aj definíciu triedy `Parameters`, ktorá parsuje argumenty príkazového riadku.
- `sentence.cpp` Obsahuje definíciu dátovej triedy `Sentence`, do ktorej sa parsujú údaje zo vstupného CSTS súboru.
- `statistics.cpp` Obsahuje štatistické funkcie, ktoré vykonávajú výpočet a zobrazenie záverečnej štatistiky analyzovaného súboru.
- `syan.cpp` Obsahuje definíciu syntaktického modulu, ktorý vykonáva syntaktickú analýzu kolokácií.
- `errors.cpp` Obsahuje definície tried `ErrorItem` a `ErrorList`, ktoré implementujú spracovanie detekovaných chybných kolokácií a ich záverečné vyhodnotenie.
- `Makefile` Makefile projektu obsahuje okrem príkazov na samotnú kompiláciu aj možnosť definovať makro `DEBUG`. Aplikácia skompilovaná s týmto makrom bude na štandardný chybový výstup vypisovať množstvo informácií o jej behu.

4.2.2 Spracovanie argumentov príkazového riadku

O spracovanie argumentov príkazového riadku sa stará trieda `Parameters` v súbore `parsing.cpp`. Obsahuje údajové premenné, ktoré reprezentujú možné prepínače aplikácie, a ktoré sa inicializujú volaním konštruktora triedy s parametrami `int` *argc* a `char**` *argv*.

V prípade chybného zadania prepínačov, konštruktor zobrazí krátku nápovedu s definíciou správneho použitia parametrov a následne ukončí aplikáciu volaním systémovej funkcie `exit(3)`. Situácia, keď konštruktor aplikáciu neukončí, nastáva jedine v prípade, ak užívateľ použil korektne všetky prepínače.

Aplikácia disponuje aj špeciálnym prepínačom `-h`. Ak konštruktor objaví medzi argumentami príkazového riadku tento prepínač, zobrazí podrobnú nápovedu o aplikácii SyMorAn a následne aplikáciu ukončí volaním `exit(3)`.

V prípade, že užívateľ použil prepínač `-m`, je volaná metóda `CheckModules()`, ktorá skontroluje parameter prepínača, ktorým je reťazec tvorený znakmi identifikujúcimi morfologické moduly. Ak v reťazci nájde znak, ktorý nepatrí žiadnemu modulu, metóda vypíše chybovú hlášku a ukončí aplikáciu volaním `exit(3)`.

Po úspešnom overení všetkých argumentov a (ak je to potrebné) reťazca s definíciou modulov, sú načítané parametre prístupné v inštancii triedy `Parameters` ako `public` premenné.

4.2.3 Predspracovanie a parsovanie vstupného súboru

Po spracovaní argumentov príkazového riadku a ich prehľadnom vypísaní na štandardný výstup aplikácia začne spracovávať vstupný CSTS súbor. Spracovanie

pozostáva z dvoch krokov. V prvom kroku sa vytvorí dočasný súbor, do ktorého sa skopírujú všetky relevantné riadky z pôvodného vstupného súboru. V druhom kroku sa z dočasného súboru parsuje postupne veta po vete a na každej sa samostatne vykonáva analýza zmysluplnosti.

Čítanie súborov v oboch krokoch a ich parsovanie vykonáva trieda `Parsing`. Pri prvom kroku (*preprocessingu*) využívame vlastnosť `tool_chainu`, ktorý CSTS súbor formátuje tak, že informácie o každom vetnom tokene sa nachádzajú na samostatnom riadku. Do dočasného súboru tak stačí uložiť len riadky začínajúce reťazcom `<s` (naznačuje začiatok novej vety a zároveň obsahuje jednoznačnú identifikáciu vety vytvorenú nástrojmi `tool_chainu`), `<f`, alebo `<d` (značia začiatok údajov o vetnom tokene). Dočasný súbor sa ukladá do adresára `/tmp`, pričom meno tohto súboru obsahuje PID spustenej aplikácie, čo umožňuje spustiť analýzu viacerých CSTS súborov súčasne. Čítanie vstupného súboru, vytvorenie dočasného súboru a jeho naplnenie údajmi je náplň práce metódy `preprocessing()`.

Druhý krok spracovania súboru už pracuje len s dočasným súborom, v ktorom sú uložené len riadky relevantné pre analýzu zmysluplnosti. Kým aplikácia neprečíta celý súbor, je v cykle `while` vykonávaný tento algoritmus:

1. Načítaj vetu
2. Analyzuj vetu a urč jej zmysluplnosť
3. Vypíš výsledok analýzy a uvoľni systémové prostriedky

Načítanie vety prebieha volaním metódy `get_sentence()`, ktorá dostáva v parametri referenciu na otvorený stream s dočasným súborom a vracia smerník na alokovaný objekt `Sentence`, v ktorom sú uložené všetky naparsované údaje.

4.2.4 Morfologické moduly

Prvým krokom pri analýze zmysluplnosti je kontrola načítanej vety morfologickými modulmi. Každý modul je implementovaný ako samostatná trieda `Modul*`, kde namiesto znaku `*` je doplnený znak identifikujúci konkrétny modul. Zoznam znakov je uvedený napríklad v tabuľke 4.2. Každý morfologický modul je zároveň potomkom abstraktnej triedy `Modules`, ktorá definuje abstraktnú virtuálnu metódu `check()`. Táto metóda volaná v konkrétnom module spustí morfologickú analýzu vety zadanej v prvom parametri. Zoznam detekovaných chýb metóda vráti v objekte triedy `ErrorList`, ktorého pointer sa metóde predáva ako druhý parameter.

V drvivej väčšine prípadov metóda check volá jednotlivé morfologické testy konkrétneho modulu, ktorým predá oba parametre a po ich ukončení vracia `true`.

Okrem deklarácie abstraktnej virtuálnej metódy `check()` trieda obsahuje aj deklarácie `public` údajových premenných spoločných pre všetky moduly. Metóda `checkModules()` triedy `Parsing` nastavuje pre každý modul premennú `usage`, ktorá nesie príznak o tom, či sa má daný modul použiť pri analýze alebo nie.

Popísaná štruktúra tried reprezentujúcich morfologické moduly umožňuje uložiť pointre na všetky moduly do jedného polymorfného poľa

```
Modules *modules[MODULES_COUNT];
```

Analýza vety všetkými modulmi sa potom deje jednoducho v cykle

```
for (i = 0; i < MODULES_COUNT; ++i)
   if (modules[i]->usage)
      modules[i]->check(*veta, zoznam_chyb);
```

Po skončení analýzy inštancia objektu `ErrorList` obsahuje práve všetky chyby detekované morfologickými modulmi.

4.2.5 Syntaktické moduly

Po ukočení morfologickej analýzy sa v cykle analyzuje každá detekovaná chyba. Chyby, ktorých druh je nastavený na tretí, sú analyzované ešte syntaktickým modulom `ModulS`, implementovaným v súbore `syan.cpp`. Metóda `check()` tohto modulu očakáva v parametroch analyzovanú vetu (v podobe objektu triedy `Sentence`) a index prvého slova v chybnej kolokácii. Metóda overí, či sú slová v kolokácii spojené hranou v závislostnom strome (či číslo otca v prvom tokene je číslo druhého tokenu alebo naopak). Ak hrana existuje, metóda vráti `true`, v opačnom prípade `false`.

Podľa výsledku metódy sa druh chyby zmení na 4, ak hrana neexistuje, alebo ostane nezmený, ak hrana existuje.

4.2.6 Vyhodnocovanie chýb

Po syntaktickej analýze je zavolaná metóda `check()` triedy `ErrorList`, ktorá zráta počty chýb podľa ich druhu a porovná ich s citlivosťami chýb predanými v parametri. Ak počet chýb aspoň jedného druhu presiahne zadanú citlivosť, veta sa vyhodnocuje ako nezmysluplná. V opačnom prípade ako zmysluplná.

Trieda `ErrorList` je implementovaná v súbore `errors.cpp`. Obsahuje predovšetkým údajovú štruktúru `vector<ErrorItem*>`, ktorá obsahuje smerníky na objekty `ErrorItem`, ktoré obsahujú údaje o kontrétnej detekovanej chybe. Pre pridanie chyby do zoznamu už detekovaných chýb sa používa metóda `add()` a pre záverečné vyhodnotenie už spomínaná metóda `check()`.

Keďže trieda `ErrorList` obsahuje smerníky na dynamicky alokované údaje, bol implementovaný aj jej deštruktor, ktorý v cykle volá deštruktor triedy `ErrorItem` na všetky smerníky uložené vo vektore.

4.2.7 Záverečná štatistika

Po spracovaní celého dočasného súboru je zavolaná funkcia `get_statistics()` v súbore `statistics.cpp`, ktorá vypočíta a vypíše záverečnú štatistiku. Funkcia používa globálne premenné `sentences_count` a `sentences_count_ok` definované v súbore `main.cpp` a premenné `ko_count`, ktoré obsahuje každá trieda morfologických modulov.

Na základe týchto premenných funkcia vypočíta percentuálne zastúpenie zmysluplných a nezmysluplných viet v súbore a percentuálny podiel morfologických modulov na detekovaní chýb.

Kapitola 5

Výsledky na testovacích údajoch

Implementovanú procedúru sme testovali na dvoch sadách viet. Prvou sadou je *vývojový* súbor viet získaných z interných zdrojov Ústavu formálnej a aplikovanej ligvistiky. Tento súbor sme mali k dispozícii pri vyvíjaní procedúry. Druhou sadou je tzv. *evaluačný* súbor viet, ktorý sme nemali k dispozícii, a test previedla vedúca práce.

Pre porovnanie sme pomocou oboch sád testovali aj aplikáciu MorAn, ktorá bola implementovaná vrámci ročníkového projektu. Výsledky aplikácie Moran boli použité ako tzv. *baseline* pre všetky ďalšie experimenty.

5.1 Vývojové údaje

Sada vývojových údajov sa skladala z dvoch súborov. Prvý z nich obsahoval 20 zmysluplných viet, druhý 101 nezmysluplných viet. Tabuľka 5.1 uvádza výsledky testovania 20 zmysluplných viet. Tabuľka 5.2 potom výsledky na súbore nezmysluplných viet.

5.2 Evaluačné údaje

Výsledky evaluačnej sady sú uvedené v tabuľkách 5.3 a 5.4

5.3 Hodnotenie výsledkov

Z výsledkov testovania vyplýva, že u aplikácie SyMorAn nenastalo výrazné zvýšenie úspešnosti. Pri zmysluplných vetách sa úspešnosť analýzy pohybuje na hranici 70 %, pri nezmysluplných vetách na úrovni 30 %.

Na týchto výsledkoch sa podieľa viacero faktorov. Jedným z nich je napríklad to, že externé moduly procedúry `tool_chain` nepracujú so stopercentnou úspešnosťou a tak môžu nastať prípady, že SyMorAn nedostane korektné dáta. Jedná sa predovšetkým o chyby taggra, ktorý z pomedzi ponúkaných možných morfologických značiek nemusí vždy vybrať tú správnu. Úspešnosť procedúry môže tiež ovplyvňovať parser, ak syntax vety analyzuje nesprávne. Aj kvôli týmto prípadom je vhodné previesť manuálne kontrolu detekovaných chýb a zo zoznamu vylúčiť chyby, ktoré boli detekované kvôli nekorektnému vstupu.

MorAn			**SyMorAn**		
Zmysluplných viet	14	70,00 %	Zmysluplných viet	14	70,00 %
Nezmysluplných viet	6	30,00 %	Nezmysluplných viet	6	30,00 %
Počet chýb spolu	-	-	Počet chýb spolu	7	-
Modul X	0	0,00 %	Modul X	0	0,00 %
Modul A	1	16,67 %	Modul A	1	14,29 %
Modul C	0	0,00 %	Modul C	0	0,00 %
Modul J	0	0,00 %	Modul J	0	0,00 %
Modul N	2	33,33 %	Modul N	2	28,57 %
Modul P	3	50,00 %	Modul P	4	57,14 %
Modul V	0	0,00 %	Modul V	0	0,00 %
Modul R	0	0,00 %	Modul R	0	0,00 %

Tabuľka 5.1: Štatistiky vývojového súboru zmysluplných viet

MorAn			**SyMorAn**		
Zmysluplných viet	70	69,31 %	Zmysluplných viet	70	69,31 %
Nezmysluplných viet	31	30,69 %	Nezmysluplných viet	31	30,69 %
Počet chýb spolu	-	-	Počet chýb spolu	39	-
Modul X	0	0,00 %	Modul X	0	0,00 %
Modul A	8	25,81 %	Modul A	9	23,08 %
Modul C	1	3,26 %	Modul C	1	2,56 %
Modul J	0	0,00 %	Modul J	0	0,00 %
Modul N	1	3,26 %	Modul N	4	10,26 %
Modul P	4	12,90 %	Modul P	8	20,51 %
Modul V	0	0,00 %	Modul V	0	0,00 %
Modul R	17	54,84 %	Modul R	17	43,59 %

Tabuľka 5.2: Štatistiky vývojového súboru nezmysluplných viet

MorAn			**SyMorAn**		
Zmysluplných viet	148	73,63 %	Zmysluplných viet	147	73,13 %
Nezmysluplných viet	53	26,37 %	Nezmysluplných viet	54	26,87 %
Počet chýb spolu	-	-	Počet chýb spolu	82	-
Modul X	9	16,98 %	Modul X	12	14,63 %
Modul A	21	39,62 %	Modul A	25	30,49 %
Modul C	0	0,00 %	Modul C	1	1,22 %
Modul J	0	0,00 %	Modul J	0	0,00 %
Modul N	13	24,53 %	Modul N	28	34,15 %
Modul P	5	9,43 %	Modul P	7	8,54 %
Modul V	0	0,00 %	Modul V	0	0,00 %
Modul R	5	9,43 %	Modul R	9	10,98 %

Tabuľka 5.3: Štatistiky evaluačného súboru zmysluplných viet

MorAn			**SyMorAn**		
Zmysluplných viet	57	81,43 %	Zmysluplných viet	57	81,43 %
Nezmysluplných viet	13	18,57 %	Nezmysluplných viet	13	18,57 %
Počet chýb spolu	-	-	Počet chýb spolu	18	-
Modul X	2	15,38 %	Modul X	2	11,11 %
Modul A	1	7,69 %	Modul A	1	5,56 %
Modul C	0	0,00 %	Modul C	0	0,00 %
Modul J	0	0,00 %	Modul J	1	5,56 %
Modul N	8	61,54 %	Modul N	11	61,11 %
Modul P	2	15,38 %	Modul P	2	11,11 %
Modul V	0	0,00 %	Modul V	0	0,00 %
Modul R	0	0,0 %	Modul R	1	5,56 %

Tabuľka 5.4: Štatistiky evaluačného súboru nezmysluplných viet

Výsledky ovplyvňuje nemalým podielom aj fakt, že súbory nezmysluplných viet neobsahujú chyby na morfologickej a syntaktickej rovine. Počas implementácie procedúry sa totiž ukázalo, že vybrať súbor nezmysluplných viet je prinajmenšom tak zložité, ako navrhnúť samotnú procedúru. V súbore nezmysluplných viet sú tak nemalým podielom zastúpené aj syntakticky zmysluplné vety.

Príkladom môže byť napríklad veta *Dali přednost jeho ženě před jím.* Jediná možná chybná kolízia by mohla nastať v dvojici *před jím.* Obe slová sú však v siedmom páde, takže morfologické moduly nedetekujú žiadnu chybu. Podobne, syntaktický strom vety sa zhoduje so stromom zmysluplnej vety *Dali přednost jeho ženě před ním.* Testovanú vetu teda môžeme chápať ako zmysluplnú vetu a v súbore nezmysluplných viet sa ocitla neprávom.

Kapitola 6

SyMorAn ako webová aplikácia

Súčasťou práce bola implementácia webovej aplikácie, ktorá by populárnym spôsobom prezentovala výsledky nášho vývoja. Stránka bola umiestnená na serveri Ústavu formálnej a aplikovanej ligvistiky MFF UK a čitateľ ju nájde na stránke `http://ufallab2.ms.mff.cuni.cz/~kriz`.

Jej cieľom bolo živou a užívateľsky nenáročnou formou umožniť prístup k výsledkom našej aplikácie každému návštevníkovi stránok. Stránka obsahuje formulár pre zadanie reťazca slov určeného na analýzu zmusluplnosti. Po jeho odoslaní zobrazí výsledky aplikácie SyMorAn v užívateľsky príjemnej podobe. Na zobrazenej vete vyznačí chybné kolokácie a popíše chybu, ktorá bola detekovaná. V opačnom prípade vypíše, že veta je zmysluplná.

Webová stránka umožňuje aj zadávanie všetkých prepínačov, ktorými SyMorAn disponuje. Sú prístupné po ťuknutí na odkaz *Pokročilé nastavenia* a užívateľ si v zobrazenom formulári môže nastaviť citlivosti jednotlivých chýb, vybrať moduly, ktoré sa použujú a zobraziť výstup SyMorAnu tak, ako je formátovaný štandardne pri *konzolovej* práci s aplikáciou.

Kapitola 7

Záver

V práci sme implementovali procedúru, ktorá v reálnom čase rozpozná, či je reťazec českých slov zmysluplná alebo nezmysluplná česká veta. V práci sa zaoberáme tzv. *syntaktickou zmysluplnosťou* a skúmame vetu pomocou dvoch rovín jazyka - morfologickej a syntaktickej. Pomocou nich sme tiež definovali samotný pojem syntaktickej zmysluplnosti.

Na záver vymenúvame niektoré dobré vlastnosti aplikácie a tiež návrhy, akým smerom by sa mohol uberať ďalší vývoj aplikácie.

7.1 Klady aplikácie

Aplikácia SyMorAn vznikala na základoch aplikácie MorAn. Prebrala preto niektoré dobré vlastnosti a vylepšila vlastnosti, ktoré sa ukázali ako nepostačujúce počas práce na ročníkovom projekte.

Medzi výhody aplikácie SyMorAn patrí, okrem iného, jej lineárna časová a pamäťová zložitosť. Dokáže v krátkom čase spracovať veľké množstvo viet určených na analýzu. Na tom sa v nemalom množstve podieľa jazyk C++, v ktorom je aplikácia napísaná a tiež fakt, že aplikácia nevykonáva žiadne diskové operácie, okrem načítania a parsovania vstupného súboru. Všetky svoje komponenty má zakompilované a dokáže ich preto používať bez zdĺhavého načítavania.

Aplikácia vyžaduje jediný povinný parameter a dá sa preto spustiť bez náročného kombinovania prepínačov. Všetky informácie píše v prehľadnom výstupe na štandardný výstup. Ten sa potom dá jednoducho, pomocou shellu, presmerovať do súboru.

K prehľadnosti výrazne prispieva aj záverečná štatistika, ktorú si program spočítava sám. Užívateľ sa tak dozvie požadované údaje bez nutnosti používania ďalších nástrojov na spracovanie výstupu (napr. programov `grep` alebo `wc`), ako tomu bolo v aplikácii MorAn.

Pri vývoji SyMorAnu sme stáli pred otázkou, či umožníme užívateľovi, aby mohol sám upravovať morfologické a syntaktické pravidlá aplikácie. Rozhodli sme sa neumožniť mu to. Predpokladali sme užívateľa, ktorý požaduje určenie zmysluplnosti a nechce sa zaoberať skúmaním kolokácií. Vďaka tomu môžu byť všetky moduly zakompilované priamo do binárneho kódu aplikácie a užívateľ nemusí čakať na ich načítavanie.

7.2 Návrhy na zlepšenie

Keďže aplikácia obsahuje nemalé množstvo rôznych prepínačov, ako vhodný nápad na zlepšenie vidíme možnosť načítavania konfiguračného súboru, ktorý by obsahoval všetky nastavenia aplikácie. Aplikácia by najskôr spracovala argumenty príkazového riadku, a ak by nejaký z prepínačov chýbal, použila by hodnotu z konfiguračného súboru.

Možným zlepšením úspešnosti procedúry by mohol byť ďalší vývoj syntaktického modulu. Ten, v súčasnej verzii, napríklad vôbec nevyužíva k rozhodovaniu o zmysluplnosti analytickú funkciu. Možné zlepšenie modulu preto vidíme práve v nájdení pravidiel pracujúcich s týmto typom informácií.

Jedným z možných smerov, kam pokračovať s vývojom syntaktickémo modulu, vidíme v kontrole *vetných skladov*, ktoré by sa dali získať zo syntaktických informácií a overiť ako samostatná veta v morfologických moduloch.

Literatúra

[1] Vidová Hladká Barbora, Jan Hajič, Jirka Hana, Jaroslava Hlaváčová, Jiří Mírovský, Jan Raab: Czech Academic Corpus 2.0. CD-ROM, ISBN: 1-58563-491-3. Linguistic Data Consortium, cat. num.: LDC2008T22, Philadelphia, Pennsylvania, USA, 2008.

[2] Czech Academic Corpus 2.0 on-line.
`http://ufal.mff.cuni.cz/rest/cac/cac_20.html`

[3] Průvodce Pražským závislostním korpusem 2.0 on-line.
`http://ufal.mff.cuni.cz/pdt2.0/`

[4] Kríž Vincent: Užívateľská dokumentácia ročníkového projektu Určovanie zmysluplnosti českej vety. Ročníkový projekt, MFF UK, Praha 2008.

Dodatok A

Morfologické moduly

V nasledujúcom prehľade sú uvedené všetky morfologické moduly a všetky ich testy, ktoré aplikácia SyMorAn obsahuje. Pri popise väčšiny pravidiel je uvedená prehľadná tabuľka, ktorá obsahuje pravidlá, ktoré testujeme medzi testovaným slovným poddruhom a za nim stojacim slovom. Na nasledujúcom príklade takejto tabuľky sú vysvetlené všetky značky, ktoré sa v tabuľke vyskytujú.

C=													
A2	o	Ch	o	Cy	o	P0	o	PH	o	VB	o	RF	o
AA	o	Cj	o	Cz	o	P1	o	PJ	o	Vc	o	RR	o
AC	o	Ck	o	C}	o	P4	o	PK	o	Ve	o	RV	o
AG	o	Cl	o	Db	o	P5	o	PL	o	Vf	o	TT	o
AM	o	Cn	o	Dg	o	P6	o	PP	o	Vi	o	ZZ	o
AO	o	Co	o	II	o	P7	o	PQ	o	Vm	o	<s>	o
AU	o	Cr	o	J*	o	P8	o	PS	o	Vp	o		
C=	o	Cu	o	J,	o	P9	o	PW	o	Vq	o		
Ca	o	Cv	o	J^	o	PD	o	PY	o	Vs	o		
Cd	o	Cw	o	NN	o	PE	o	PZ	o	Vt	o		

Vzorová tabuľka zobrazuje pravidlá pre dvojicu slov, kde prvá z dvojice je číslovka písaná číslom (morfologická značka tohto slovného poddruhu začína znakmi C=). V stĺpcoch potom nasleduje zoznam možných slovných poddruhov a v susednom stĺpci potom značka vyjadrujúca pravidlo.

Okrem morfologických značiek je v tabuľke uvedená aj značka <s>, ktorá vyjadruje, či sa daný slovný poddruh môže vyskytovať na konci vety.

Pravidlá sú vyjadrené niekoľkými symbolmi, ich prehľad uvádzame v tabuľke A.1. Špeciálnymi znakmi uvádzanými v tabuľke sú čísla. Vyjadrujú zhodu medzi testovanými tokenmi na danom znaku v morfologickej značke. Prehľad jednotlivých znakov je uvedený napríklad v [3].

A.1 X - neklasifikovateľné slová

Modul X obsahuje jediný test, ktorý testuje vetu na výskyt znažiek X@----------. Takáto chyba je ohodnotená ako chyba prvého druhu.

Symbol	Popis pravidla	Druh chyby
o	Slovný poddruh môže nasledovať, nie je známe presnejšie pravidlo	-
x	Slovný poddruh nemôže nasledovať	2
345	Slovný poddruh sa musí zhodovať v rode, čísle a páde	3
P5	Slovný poddruh v pluráli sa musí zhodovať v páde	3
S5	Slovný poddruh v singulári sa musí zhodovať v páde	3
*5	Slovný poddruh sa musí zhodovať v páde, ale len za špeciálnej podmienky vyjadrenej mimo tabuľku	3

Tabuľka A.1: Symboly použité v morfologických tabuľkách

A.2 R - predložky

Modul R obsahuje tri testy, ktoré kontrolujú korektnosť predložiek vyskytujúcich sa v danej vete. Nižšie je uvedený ich podrobný popis.

Test RF

Tagom `RF----------` sú označované tzv. nepravé predložky - slová, ktoré nikdy nestoja osamote a vždy po nich nasleduje predložka, napríklad *vzhledem (k), nehledě (na)*. Test skontroluje, či po každom z týchto slov predložka skutočne nasleduje.

Test RZ

Predložky vo väčšine prípadov rozvíjajú slovný druh, ktorý sa skloňuje a nasleduje za predložkou. Je málo pravdepodobné, ak sa predložka nachádza na konci vety. Test tento kontroluje práve tento jav.

Program označí detekuje chybu prvého druhu pre vety s predložkou na konci vety alebo časti súvetia.

Test nasledovníkov

Predložky rozvíjajú za nimi stojaci slovný druh, ktorý sa skloňuje. Navyše sa viažu s pádom. Tento test sleduje, aký slovný druh nasleduje za predložkou. V nasledujúcej tabuľke je uvedený prehľad, akým spôsobom procedúra ďalej spracúva pár predložka a za ňou stojaci slovný poddruh.

RR, RV													
A2	o	Ch	5	Cy	5	P0	x	PH	5	VB	x	RF	x
AA	5	Cj	5	Cz	5	P1	5	PJ	o	Vc	x	RR	o
AC	o	Ck	5	C}	o	P4	5	PK	5	Ve	x	RV	o
AG	5	Cl	5	Db	o	P5	5	PL	5	Vf	x	TT	x
AM	5	Cn	5	Dg	o	P6	5	PP	5	Vi	x	ZZ	x
AO	x	Co	o	II	o	P7	o	PQ	5	Vm	x	<s>	x
AU	5	Cr	5	J*	o	P8	5	PS	5	Vp	x		
C=	o	Cu	x	J,	o	P9	5	PW	5	Vq	x		
Ca	5	Cv	o	J^	o	PD	5	PY	o	Vs	x		
Cd	5	Cw	5	NN	5	PE	5	PZ	5	Vt	x		

V málo pravdepodobných prípadoch môžu nasledovať dve predložky za sebou. Takáto vetná konštrukcia je však málo pravdepodobná a program ju označí za chybnú.

A.3 A - prídavné mená

Modul obsahuje testy, ktoré kontrolujú prídavné mená. Obsahuje samostatný test pre každý slovný poddruh.

Test A2

Test kontroluje špecifické prídavné mená so značkou `A2----------`. Ide o prídavné mená spojené s ďalším prídavným menom pomocou spojovníka a vyparsované ako samostatný token, napríklad *technicko-*. Test kontroluje, či za každým z takýchto prídavných mien nasleduje ďalšie prídavné meno.

Test AA

AA													
A2	o	Ch	o	Cy	345	P0	o	PH	o	VB	4	RF	o
AA	5	Cj	5	Cz	x	P1	o	PJ	o	Vc	4	RR	o
AC	o	Ck	5	C}	o	P4	o	PK	o	Ve	o	RV	o
AG	5	Cl	o	Db	o	P5	x	PL	o	Vf	o	TT	o
AM	5	Cn	o	Dg	o	P6	o	PP	o	Vi	4	ZZ	o
AO	4	Co	o	II	o	P7	o	PQ	o	Vm	o	<s>	o
AU	5	Cr	345	J*	o	P8	5*	PS	o	Vp	o		
C=	o	Cu	o	J,	o	P9	x	PW	o	Vq	x		
Ca	o	Cv	o	J^	o	PD	o	PY	o	Vs	o		
Cd	5	Cw	o	NN	345	PE	o	PZ	o	Vt	x		

* Privlastňovacie zámená `P8` sa musia zhodovať v páde, len ak je pred `AA` predložka.

Test AC

AC													
A2	o	Ch	o	Cy	o	P0	o	PH	o	VB	o	RF	o
AA	o	Cj	o	Cz	o	P1	o	PJ	o	Vc	o	RR	o
AC	o	Ck	o	C}	o	P4	o	PK	o	Ve	o	RV	o
AG	o	Cl	o	Db	o	P5	x	PL	o	Vf	o	TT	o
AM	x	Cn	o	Dg	o	P6	o	PP	o	Vi	o	ZZ	o
AO	x	Co	o	II	o	P7	o	PQ	o	Vm	o	<s>	o
AU	o	Cr	o	J*	x	P8	o	PS	o	Vp	o		
C=	o	Cu	o	J,	o	P9	x	PW	o	Vq	x		
Ca	o	Cv	o	J^	o	PD	o	PY	o	Vs	o		
Cd	o	Cw	o	NN	o	PE	o	PZ	o	Vt	x		

Test AG

AG													
A2	o	Ch	o	Cy	o	P0	o	PH	o	VB	o	RF	o
AA	o	Cj	o	Cz	o	P1	o	PJ	o	Vc	o	RR	o
AC	o	Ck	o	C}	o	P4	o	PK	o	Ve	o	RV	o
AG	o	Cl	o	Db	o	P5	x	PL	o	Vf	o	TT	o
AM	o	Cn	o	Dg	o	P6	o	PP	o	Vi	o	ZZ	o
AO	o	Co	o	II	o	P7	o	PQ	o	Vm	o	<s>	o
AU	o	Cr	o	J*	x	P8	5*	PS	o	Vp	o		
C=	o	Cu	o	J,	o	P9	x	PW	o	Vq	x		
Ca	o	Cv	o	J^	o	PD	o	PY	o	Vs	o		
Cd	o	Cw	o	NN	345	PE	o	PZ	o	Vt	x		

Test AM

AM													
A2	o	Ch	o	Cy	o	P0	o	PH	o	VB	o	RF	o
AA	o	Cj	o	Cz	o	P1	o	PJ	o	Vc	o	RR	o
AC	x	Ck	o	C}	o	P4	o	PK	o	Ve	o	RV	o
AG	x	Cl	o	Db	o	P5	x	PL	o	Vf	o	TT	o
AM	x	Cn	o	Dg	o	P6	o	PP	o	Vi	o	ZZ	o
AO	x	Co	o	II	o	P7	o	PQ	o	Vm	o	<s>	o
AU	o	Cr	o	J*	x	P8	o	PS	o	Vp	o		
C=	o	Cu	o	J,	o	P9	x	PW	o	Vq	x		
Ca	o	Cv	o	J^	o	PD	o	PY	o	Vs	o		
Cd	o	Cw	o	NN	345	PE	o	PZ	o	Vt	x		

Test AO

AO													
A2	x	Ch	x	Cy	x	P0	x	PH	x	VB	o	RF	o
AA	x	Cj	x	Cz	x	P1	x	PJ	x	Vc	o	RR	o
AC	x	Ck	x	C}	x	P4	x	PK	x	Ve	o	RV	o
AG	x	Cl	x	Db	o	P5	x	PL	x	Vf	o	TT	o
AM	x	Cn	x	Dg	o	P6	o	PP	x	Vi	o	ZZ	o
AO	x	Co	x	II	o	P7	o	PQ	x	Vm	o	<s>	o
AU	x	Cr	x	J*	o	P8	x	PS	x	Vp	o		
C=	x	Cu	x	J,	o	P9	x	PW	x	Vq	x		
Ca	x	Cv	x	J^	o	PD	x	PY	x	Vs	o		
Cd	x	Cw	x	NN	o	PE	x	PZ	x	Vt	x		

Test AU

AU													
A2	o	Ch	o	Cy	5	P0	x	PH	o	VB	o	RF	o
AA	5	Cj	o	Cz	o	P1	x	PJ	o	Vc	o	RR	o
AC	o	Ck	o	C}	o	P4	x	PK	o	Ve	o	RV	o
AG	5	Cl	o	Db	o	P5	x	PL	o	Vf	o	TT	o
AM	5	Cn	o	Dg	o	P6	o	PP	o	Vi	o	ZZ	o
AO	o	Co	o	II	o	P7	o	PQ	o	Vm	o	<s>	o
AU	5	Cr	5	J*	x	P8	o	PS	o	Vp	o		
C=	o	Cu	o	J,	o	P9	x	PW	o	Vq	x		
Ca	o	Cv	o	J^	o	PD	o	PY	o	Vs	o		
Cd	5	Cw	o	NN	345	PE	o	PZ	o	Vt	x		

A.4 C - číslovky

Modul obsahuje testy, ktoré kontrolujú číslovky. Obsahuje samostatný test pre každý slovný poddruh.

Test čísel

Tento test sa zameriava na číslovky napísané číslom. Značka popisujúca čísla neobsahuje žiadnu ďalšiu morfologickú informáciu. Test však využíva to, že pri číslach môžeme z ich hodnoty určiť, či ide o jednotné alebo množné číslo. Bohužiaľ, túto gramatickú informáciu nemožno využiť hromadne a je známy len jeden prípad kedy ju využijeme.

Ide o prípad, ak za číslom nasleduje číslovka vyjadrujúca zlomok (značka `Cy`). V tomto prípade, ak je číslo väčšie ako 1, musí byť zlomok v množnom čísle, inak v jednotnom.

Test následne kontroluje, aký slovný poddruh môže za číslom nasledovať a aký nie. Súčasťou testu je aj kontrola, či sa veta neskladá len z tohto slovného druhu.

C=												
A2	o	Ch	o	Cy	4	P0	o	PH	o	VB	o	RF	o
AA	o	Cj	x	Cz	x	P1	o	PJ	o	Vc	o	RR	o
AC	o	Ck	x	C}	o	P4	o	PK	x	Ve	o	RV	o
AG	o	Cl	o	Db	o	P5	x	PL	o	Vf	o	TT	o
AM	o	Cn	o	Dg	o	P6	o	PP	o	Vi	o	ZZ	o
AO	o	Co	o	II	o	P7	o	PQ	o	Vm	o	<s>	o
AU	o	Cr	o	J*	o	P8	o	PS	o	Vp	o		
C=	o	Cu	x	J,	o	P9	x	PW	o	Vq	x		
Ca	o	Cv	o	J^	o	PD	o	PY	x	Vs	o		
Cd	o	Cw	o	NN	o	PE	o	PZ	o	Vt	x		

Test Ca

Ca												
A2	o	Ch	x	Cy	o	P0	x	PH	o	VB	o	RF	o
AA	o	Cj	x	Cz	x	P1	x	PJ	x	Vc	o	RR	o
AC	o	Ck	x	C}	o	P4	o	PK	o	Ve	x	RV	o
AG	o	Cl	o	Db	o	P5	x	PL	o	Vf	o	TT	o
AM	o	Cn	o	Dg	o	P6	o	PP	o	Vi	o	ZZ	o
AO	x	Co	o	II	o	P7	o	PQ	o	Vm	o	<s>	o
AU	o	Cr	o	J*	x	P8	o	PS	o	Vp	o		
C=	o	Cu	x	J,	o	P9	x	PW	o	Vq	x		
Ca	5	Cv	o	J^	o	PD	o	PY	x	Vs	o		
Cd	o	Cw	x	NN	o	PE	o	PZ	o	Vt	x		

Test Cd

Cd												
A2	o	Ch	x	Cy	5	P0	x	PH	o	VB	o	RF	x
AA	5	Cj	x	Cz	o	P1	x	PJ	x	Vc	o	RR	o
AC	x	Ck	x	C}	o	P4	x	PK	x	Ve	o	RV	o
AG	5	Cl	x	Db	o	P5	x	PL	x	Vf	o	TT	o
AM	x	Cn	x	Dg	o	P6	x	PP	o	Vi	o	ZZ	o
AO	x	Co	o	II	o	P7	o	PQ	x	Vm	o	<s>	o
AU	5	Cr	5	J*	x	P8	o	PS	o	Vp	o		
C=	o	Cu	x	J,	o	P9	x	PW	x	Vq	x		
Ca	x	Cv	o	J^	o	PD	5	PY	x	Vs	o		
Cd	5	Cw	x	NN	5	PE	x	PZ	o	Vt	x		

Test Ch

Ch													
A2	o	Ch	x	Cy	o	P0	o	PH	o	VB	o	RF	x
AA	o	Cj	x	Cz	x	P1	x	PJ	x	Vc	o	RR	o
AC	o	Ck	x	C}	o	P4	x	PK	x	Ve	o	RV	o
AG	o	Cl	o	Db	o	P5	x	PL	o	Vf	o	TT	o
AM	x	Cn	o	Dg	o	P6	o	PP	o	Vi	o	ZZ	o
AO	x	Co	o	II	x	P7	o	PQ	x	Vm	o	<s>	o
AU	o	Cr	o	J*	x	P8	o	PS	o	Vp	o		
C=	o	Cu	x	J,	o	P9	x	PW	o	Vq	x		
Ca	o	Cv	o	J^	o	PD	o	PY	x	Vs	o		
Cd	o	Cw	x	NN	o	PE	x	PZ	o	Vt	x		

Test Cj

Cj													
A2	o	Ch	x	Cy	o	P0	o	PH	o	VB	o	RF	o
AA	o	Cj	x	Cz	x	P1	o	PJ	o	Vc	o	RR	o
AC	x	Ck	x	C}	o	P4	o	PK	x	Ve	o	RV	o
AG	o	Cl	o	Db	o	P5	x	PL	o	Vf	o	TT	o
AM	x	Cn	o	Dg	o	P6	o	PP	o	Vi	o	ZZ	o
AO	x	Co	o	II	x	P7	o	PQ	x	Vm	o	<s>	o
AU	o	Cr	o	J*	o	P8	o	PS	o	Vp	o		
C=	o	Cu	x	J,	o	P9	x	PW	o	Vq	x		
Ca	x	Cv	o	J^	o	PD	o	PY	x	Vs	o		
Cd	x	Cw	x	NN	o	PE	x	PZ	o	Vt	x		

Test Ck

Ck													
A2	o	Ch	x	Cy	5	P0	o	PH	o	VB	o	RF	x
AA	5	Cj	x	Cz	x	P1	x	PJ	o	Vc	o	RR	o
AC	x	Ck	x	C}	o	P4	x	PK	x	Ve	o	RV	o
AG	5	Cl	o	Db	o	P5	x	PL	o	Vf	o	TT	o
AM	x	Cn	o	Dg	o	P6	o	PP	o	Vi	o	ZZ	o
AO	x	Co	o	II	x	P7	o	PQ	o	Vm	o	<s>	o
AU	5	Cr	x	J*	x	P8	o	PS	o	Vp	o		
C=	o	Cu	x	J,	x	P9	x	PW	o	Vq	x		
Ca	x	Cv	o	J^	o	PD	o	PY	x	Vs	o		
Cd	x	Cw	x	NN	5	PE	o	PZ	o	Vt	x		

Test Cn

Cl, Cn													
A2	o	Ch	x	Cy	5	P0	x	PH	o	VB	o	RF	x
AA	5	Cj	x	Cz	x	P1	x	PJ	o	Vc	o	RR	o
AC	o	Ck	x	C}	o	P4	o	PK	o	Ve	o	RV	o
AG	5	Cl	o	Db	o	P5	x	PL	o	Vf	o	TT	o
AM	5	Cn	5	Dg	o	P6	o	PP	o	Vi	o	ZZ	o
AO	x	Co	o	II	o	P7	o	PQ	o	Vm	o	<s>	o
AU	5	Cr	o	J*	o	P8	o	PS	o	Vp	o		
C=	o	Cu	x	J,	o	P9	x	PW	o	Vq	x		
Ca	o	Cv	o	J^	o	PD	S5	PY	o	Vs	o		
Cd	o	Cw	o	NN	o	PE	x	PZ	o	Vt	x		

Test Co

Co													
A2	o	Ch	x	Cy	o	P0	o	PH	o	VB	o	RF	o
AA	o	Cj	o	Cz	x	P1	x	PJ	o	Vc	o	RR	o
AC	o	Ck	o	C}	o	P4	x	PK	x	Ve	o	RV	o
AG	o	Cl	o	Db	o	P5	x	PL	o	Vf	o	TT	o
AM	o	Cn	o	Dg	o	P6	o	PP	o	Vi	o	ZZ	o
AO	o	Co	x	II	o	P7	o	PQ	o	Vm	o	<s>	o
AU	o	Cr	o	J*	o	P8	o	PS	o	Vp	o		
C=	o	Cu	x	J,	o	P9	x	PW	o	Vq	x		
Ca	o	Cv	o	J^	o	PD	o	PY	o	Vs	o		
Cd	o	Cw	o	NN	o	PE	x	PZ	o	Vt	x		

Test Cr

Cr													
A2	o	Ch	x	Cy	5	P0	o	PH	o	VB	o	RF	o
AA	345	Cj	x	Cz	x	P1	o	PJ	x	Vc	o	RR	o
AC	o	Ck	x	C}	o	P4	345	PK	5	Ve	o	RV	o
AG	345	Cl	P5	Db	o	P5	x	PL	o	Vf	o	TT	o
AM	345	Cn	P5	Dg	o	P6	o	PP	o	Vi	o	ZZ	o
AO	x	Co	o	II	o	P7	o	PQ	o	Vm	o	<s>	o
AU	345	Cr	345	J*	x	P8	o	PS	o	Vp	o		
C=	o	Cu	x	J,	o	P9	x	PW	o	Vq	x		
Ca	5	Cv	o	J^	o	PD	o	PY	o	Vs	o		
Cd	x	Cw	o	NN	345	PE	o	PZ	o	Vt	x		

Test Cu

Cu													
A2	o	Ch	o	Cy	o	P0	o	PH	0	VB	o	RF	x
AA	o	Cj	o	Cz	x	P1	x	PJ	x	Vc	o	RR	o
AC	o	Ck	o	C}	o	P4	o	PK	x	Ve	o	RV	o
AG	o	Cl	o	Db	o	P5	x	PL	o	Vf	o	TT	o
AM	x	Cn	o	Dg	o	P6	o	PP	o	Vi	o	ZZ	o
AO	o	Co	x	II	o	P7	o	PQ	x	Vm	o	<s>	o
AU	o	Cr	o	J*	x	P8	o	PS	o	Vp	o		
C=	o	Cu	x	J,	o	P9	x	PW	o	Vq	x		
Ca	o	Cv	x	J^	o	PD	o	PY	x	Vs	o		
Cd	o	Cw	x	NN	o	PE	x	PZ	o	Vt	x		

Test Cv

Cv													
A2	o	Ch	x	Cy	o	P0	o	PH	o	VB	o	RF	o
AA	o	Cj	x	Cz	x	P1	o	PJ	o	Vc	o	RR	o
AC	o	Ck	x	C}	o	P4	o	PK	x	Ve	o	RV	o
AG	o	Cl	o	Db	o	P5	x	PL	o	Vf	o	TT	o
AM	x	Cn	o	Dg	o	P6	o	PP	o	Vi	o	ZZ	o
AO	x	Co	x	II	o	P7	o	PQ	o	Vm	o	<s>	o
AU	o	Cr	o	J*	o	P8	o	PS	o	Vp	o		
C=	o	Cu	x	J,	o	P9	x	PW	o	Vq	x		
Ca	o	Cv	o	J^	o	PD	o	PY	x	Vs	o		
Cd	o	Cw	x	NN	o	PE	o	PZ	o	Vt	x		

Test Cw

Cw													
A2	o	Ch	x	Cy	o	P0	o	PH	o	VB	o	RF	o
AA	345	Cj	o	Cz	x	P1	o	PJ	o	Vc	o	RR	o
AC	o	Ck	o	C}	o	P4	0	PK	o	Ve	o	RV	o
AG	o	Cl	o	Db	o	P5	x	PL	o	Vf	o	TT	o
AM	x	Cn	o	Dg	o	P6	o	PP	345	Vi	o	ZZ	o
AO	x	Co	o	II	o	P7	o	PQ	o	Vm	o	<s>	o
AU	o	Cr	o	J*	o	P8	345	PS	345	Vp	o		
C=	o	Cu	x	J,	o	P9	x	PW	o	Vq	x		
Ca	o	Cv	o	J^	o	PD	345	PY	o	Vs	o		
Cd	x	Cw	x	NN	345	PE	x	PZ	o	Vt	x		

Test Cy

Cy													
A2	o	Ch	x	Cy	o	P0	o	PH	o	VB	o	RF	o
AA	o	Cj	x	Cz	x	P1	o	PJ	o	Vc	o	RR	o
AC	o	Ck	x	C}	o	P4	o	PK	x	Ve	o	RV	o
AG	o	Cl	o	Db	o	P5	x	PL	o	Vf	o	TT	o
AM	o	Cn	o	Dg	o	P6	o	PP	o	Vi	o	ZZ	o
AO	o	Co	o	II	o	P7	o	PQ	o	Vm	o	<s>	o
AU	o	Cr	o	J*	o	P8	o	PS	o	Vp	o		
C=	o	Cu	x	J,	o	P9	x	PW	o	Vq	x		
Ca	o	Cv	o	J^	0	PD	o	PY	o	Vs	o		
Cd	x	Cw	x	NN	o	PE	o	PZ	o	Vt	x		

Test Cz

Cz													
A2	o	Ch	x	Cy	o	P0	o	PH	o	VB	o	RF	o
AA	o	Cj	x	Cz	x	P1	o	PJ	o	Vc	o	RR	o
AC	o	Ck	x	C}	o	P4	o	PK	x	Ve	o	RV	o
AG	0	Cl	x	Db	o	P5	x	PL	o	Vf	o	TT	o
AM	o	Cn	x	Dg	o	P6	o	PP	o	Vi	o	ZZ	o
AO	o	Co	X	II	x	P7	o	PQ	o	Vm	o	<s>	o
AU	o	Cr	x	J*	x	P8	o	PS	o	Vp	o		
C=	o	Cu	x	J,	o	P9	x	PW	o	Vq	x		
Ca	o	Cv	x	J^	0	PD	o	PY	o	Vs	o		
Cd	x	Cw	x	NN	0	PE	x	PZ	o	Vt	x		

A.5 N - podstatné mená

Podstatné mená nie sú rozdelené do poddruhov. Procedúra preto uvažuje o všetkých podstatných menách v jednom teste.

NN													
A2	o	Ch	o	Cy	o	P0	o	PH	o	VB	o	RF	o
AA	o	Cj	o	Cz	o	P1	4	PJ	o	Vc	o	RR	o
AC	o	Ck	o	C}	o	P4	34	PK	o	Ve	o	RV	o
AG	5*	Cl	o	Db	o	P5	x	PL	o	Vf	o	TT	o
AM	o	Cn	o	Dg	o	P6	o	PP	o	Vi	o	ZZ	o
AO	o	Co	o	II	o	P7	o	PQ	o	Vm	o	<s>	o
AU	o	Cr	o	J*	o	P8	o	PS	o	Vp	o		
C=	o	Cu	o	J,	o	P9	x	PW	o	Vq	x		
Ca	o	Cv	o	J^	o	PD	o	PY	o	Vs	o		
Cd	o	Cw	o	NN	5**	PE	o	PZ	o	Vt	x		

* Zhoda v páde sa pri adjektíve odvodenom od prítomného prechodníku kontroluje, len ak pred podstatným menom nie je predložka.

** Podstatné meno, ktoré nasleduje za podstatným menom musí byť v druhom páde. (Okrem situácie, ak za dvojicou nasleduje sloveso.)

A.6 J - predložky

V tomto module je implementovaný jediný test, ktorý kontroluje, či sa spojka (podraďovacia alebo priraďovacia) nenachádza na konci vety.

A.7 P - zámená

Zámená procedúra kontroluje, podobne ako v prípade čísloviek, v samostatných testoch, každý pre jeden slovný poddruh. Okrem toho, modul obsahuje ešte test, ktorý sleduje, či sa určité druhy zámen nenachádzajú na konci vety.

Test P0

P0													
A2	o	Ch	o	Cy	o	P0	x	PH	o	VB	o	RF	o
AA	o	Cj	o	Cz	o	P1	o	PJ	o	Vc	o	RR	o
AC	o	Ck	o	C}	o	P4	o	PK	o	Ve	o	RV	o
AG	o	Cl	o	Db	o	P5	x	PL	o	Vf	o	TT	o
AM	o	Cn	o	Dg	o	P6	o	PP	o	Vi	o	ZZ	o
AO	o	Co	o	II	o	P7	o	PQ	o	Vm	o	<s>	o
AU	o	Cr	o	J*	o	P8	o	PS	o	Vp	o		
C=	o	Cu	o	J,	o	P9	x	PW	o	Vq	x		
Ca	o	Cv	o	J^	o	PD	o	PY	o	Vs	o		
Cd	o	Cw	o	NN	o	PE	o	PZ	o	Vt	x		

Test P1

P1													
A2	o	Ch	o	Cy	o	P0	o	PH	o	VB	o	RF	x
AA	o	Cj	o	Cz	o	P1	x	PJ	o	Vc	o	RR	o
AC	o	Ck	o	C}	o	P4	x	PK	x	Ve	o	RV	o
AG	o	Cl	o	Db	o	P5	x	PL	o	Vf	o	TT	o
AM	o	Cn	o	Dg	o	P6	x	PP	o	Vi	o	ZZ	o
AO	o	Co	o	II	x	P7	o	PQ	o	Vm	o	<s>	x
AU	o	Cr	o	J*	x	P8	x	PS	o	Vp	o		
C=	o	Cu	o	J,	o	P9	x	PW	o	Vq	x		
Ca	o	Cv	o	J^	o	PD	o	PY	x	Vs	o		
Cd	o	Cw	o	NN	o	PE	x	PZ	o	Vt	x		

Test P4

P4													
A2	o	Ch	o	Cy	o	P0	o	PH	o	VB	o	RF	o
AA	o	Cj	o	Cz	o	P1	o	PJ	o	Vc	o	RR	o
AC	o	Ck	o	C}	o	P4	o	PK	o	Ve	o	RV	o
AG	o	Cl	o	Db	o	P5	x	PL	o	Vf	o	TT	o
AM	o	Cn	o	Dg	o	P6	o	PP	o	Vi	o	ZZ	o
AO	o	Co	o	II	o	P7	o	PQ	o	Vm	o	<s>	o
AU	o	Cr	o	J*	x	P8	o	PS	o	Vp	o		
C=	o	Cu	o	J,	o	P9	x	PW	o	Vq	x		
Ca	o	Cv	o	J^	o	PD	o	PY	o	Vs	o		
Cd	o	Cw	o	NN	o	PE	x	PZ	o	Vt	x		

Test P5

P5													
A2	o	Ch	o	Cy	o	P0	o	PH	o	VB	o	RF	o
AA	o	Cj	o	Cz	o	P1	o	PJ	o	Vc	o	RR	o
AC	o	Ck	o	C}	o	P4	o	PK	o	Ve	o	RV	o
AG	o	Cl	o	Db	o	P5	x	PL	o	Vf	o	TT	o
AM	o	Cn	o	Dg	o	P6	o	PP	o	Vi	o	ZZ	o
AO	o	Co	o	II	o	P7	o	PQ	o	Vm	o	<s>	o
AU	o	Cr	o	J*	o	P8	o	PS	o	Vp	o		
C=	o	Cu	o	J,	o	P9	x	PW	o	Vq	x		
Ca	o	Cv	o	J^	o	PD	o	PY	o	Vs	o		
Cd	o	Cw	o	NN	o	PE	x	PZ	o	Vt	x		

Test P6

P6													
A2	o	Ch	o	Cy	o	P0	o	PH	o	VB	o	RF	o
AA	o	Cj	o	Cz	o	P1	o	PJ	o	Vc	o	RR	o
AC	o	Ck	o	C}	o	P4	o	PK	o	Ve	o	RV	o
AG	o	Cl	o	Db	o	P5	x	PL	o	Vf	o	TT	o
AM	o	Cn	o	Dg	o	P6	x	PP	o	Vi	o	ZZ	o
AO	o	Co	o	II	o	P7	o	PQ	o	Vm	o	<s>	o
AU	o	Cr	o	J*	x	P8	o	PS	o	Vp	o		
C=	o	Cu	o	J,	o	P9	x	PW	o	Vq	x		
Ca	o	Cv	o	J^	o	PD	o	PY	o	Vs	o		
Cd	o	Cw	o	NN	o	PE	o	PZ	o	Vt	x		

Test P7

P7													
A2	o	Ch	o	Cy	o	P0	o	PH	o	VB	o	RF	o
AA	o	Cj	o	Cz	o	P1	x	PJ	o	Vc	o	RR	o
AC	o	Ck	o	C}	o	P4	o	PK	o	Ve	o	RV	o
AG	o	Cl	o	Db	o	P5	x	PL	o	Vf	o	TT	o
AM	o	Cn	o	Dg	o	P6	o	PP	o	Vi	o	ZZ	o
AO	o	Co	o	II	o	P7	o	PQ	o	Vm	o	<s>	o
AU	o	Cr	o	J*	o	P8	o	PS	o	Vp	o		
C=	o	Cu	o	J,	o	P9	x	PW	o	Vq	x		
Ca	o	Cv	o	J^	o	PD	o	PY	o	Vs	o		
Cd	o	Cw	o	NN	o	PE	o	PZ	o	Vt	x		

Test P8

P8													
A2	o	Ch	o	Cy	o	P0	o	PH	o	VB	o	RF	o
AA	345	Cj	o	Cz	o	P1	x	PJ	o	Vc	o	RR	o
AC	o	Ck	o	C}	o	P4	x	PK	o	Ve	o	RV	o
AG	345	Cl	o	Db	o	P5	x	PL	o	Vf	o	TT	o
AM	345	Cn	o	Dg	o	P6	x	PP	o	Vi	o	ZZ	o
AO	x	Co	o	II	o	P7	o	PQ	o	Vm	o	<s>	o
AU	345	Cr	o	J*	o	P8	o	PS	o	Vp	o		
C=	o	Cu	o	J,	o	P9	x	PW	o	Vq	x		
Ca	o	Cv	o	J^	o	PD	345	PY	o	Vs	o		
Cd	o	Cw	o	NN	345	PE	x	PZ	o	Vt	x		

Test P9

P9													
A2	o	Ch	o	Cy	o	P0	o	PH	o	VB	o	RF	o
AA	o	Cj	o	Cz	o	P1	o	PJ	o	Vc	o	RR	o
AC	o	Ck	o	C}	o	P4	o	PK	o	Ve	o	RV	o
AG	o	Cl	o	Db	o	P5	x	PL	o	Vf	o	TT	o
AM	o	Cn	o	Dg	o	P6	o	PP	o	Vi	o	ZZ	o
AO	o	Co	o	II	o	P7	o	PQ	o	Vm	o	<s>	o
AU	o	Cr	o	J*	o	P8	o	PS	o	Vp	o		
C=	o	Cu	o	J,	o	P9	x	PW	o	Vq	x		
Ca	o	Cv	o	J^	o	PD	o	PY	o	Vs	o		
Cd	o	Cw	o	NN	o	PE	x	PZ	o	Vt	x		

Test PD

PD													
A2	o	Ch	o	Cy	o	P0	o	PH	o	VB	o	RF	o
AA	345	Cj	o	Cz	o	P1	345	PJ	o	Vc	o	RR	o
AC	o	Ck	o	C}	o	P4	345	PK	o	Ve	o	RV	o
AG	345	Cl	o	Db	o	P5	x	PL	345	Vf	o	TT	o
AM	345	Cn	o	Dg	o	P6	o	PP	o	Vi	o	ZZ	o
AO	o	Co	o	II	o	P7	o	PQ	o	Vm	o	<s>	o
AU	345	Cr	o	J*	o	P8	o	PS	345	Vp	o		
C=	o	Cu	o	J,	o	P9	x	PW	o	Vq	x		
Ca	o	Cv	o	J^	o	PD	o	PY	o	Vs	o		
Cd	o	Cw	o	NN	345	PE	o	PZ	o	Vt	x		

Test PE

PE													
A2	o	Ch	o	Cy	o	P0	o	PH	o	VB	o	RF	o
AA	o	Cj	o	Cz	o	P1	x	PJ	x	Vc	o	RR	o
AC	o	Ck	o	C}	o	P4	x	PK	o	Ve	o	RV	o
AG	o	Cl	o	Db	o	P5	x	PL	o	Vf	o	TT	o
AM	o	Cn	o	Dg	o	P6	o	PP	o	Vi	o	ZZ	o
AO	o	Co	o	II	o	P7	o	PQ	x	Vm	o	<s>	o
AU	o	Cr	o	J*	o	P8	o	PS	o	Vp	o		
C=	o	Cu	o	J,	o	P9	x	PW	o	Vq	x		
Ca	o	Cv	o	J^	o	PD	o	PY	x	Vs	o		
Cd	o	Cw	o	NN	o	PE	x	PZ	o	Vt	x		

Test PH

PH													
A2	o	Ch	o	Cy	o	P0	o	PH	o	VB	o	RF	o
AA	o	Cj	o	Cz	o	P1	x	PJ	o	Vc	o	RR	o
AC	o	Ck	o	C}	o	P4	o	PK	o	Ve	o	RV	o
AG	o	Cl	o	Db	o	P5	x	PL	o	Vf	o	TT	o
AM	o	Cn	o	Dg	o	P6	o	PP	o	Vi	o	ZZ	o
AO	o	Co	o	II	o	P7	o	PQ	o	Vm	o	<s>	o
AU	o	Cr	o	J*	o	P8	o	PS	o	Vp	o		
C=	o	Cu	o	J,	o	P9	x	PW	o	Vq	x		
Ca	o	Cv	o	J^	o	PD	o	PY	o	Vs	o		
Cd	o	Cw	o	NN	o	PE	o	PZ	o	Vt	x		

Test PJ

PJ													
A2	o	Ch	o	Cy	o	P0	o	PH	o	VB	o	RF	o
AA	o	Cj	o	Cz	o	P1	x	PJ	x	Vc	o	RR	o
AC	o	Ck	o	C}	o	P4	x	PK	o	Ve	o	RV	o
AG	o	Cl	o	Db	o	P5	x	PL	o	Vf	o	TT	o
AM	o	Cn	o	Dg	o	P6	o	PP	o	Vi	o	ZZ	o
AO	o	Co	o	II	o	P7	o	PQ	o	Vm	o	<s>	x
AU	o	Cr	o	J*	o	P8	o	PS	o	Vp	o		
C=	o	Cu	o	J,	o	P9	x	PW	o	Vq	x		
Ca	o	Cv	o	J^	o	PD	o	PY	o	Vs	o		
Cd	o	Cw	o	NN	o	PE	x	PZ	o	Vt	x		

Test PK

PK													
A2	o	Ch	o	Cy	o	P0	o	PH	o	VB	o	RF	o
AA	o	Cj	o	Cz	o	P1	x	PJ	o	Vc	o	RR	o
AC	o	Ck	o	C}	o	P4	x	PK	o	Ve	o	RV	o
AG	o	Cl	o	Db	o	P5	x	PL	o	Vf	o	TT	o
AM	o	Cn	o	Dg	o	P6	o	PP	o	Vi	o	ZZ	o
AO	o	Co	o	II	o	P7	o	PQ	o	Vm	o	<s>	o
AU	o	Cr	o	J*	o	P8	o	PS	o	Vp	o		
C=	o	Cu	o	J,	o	P9	x	PW	o	Vq	x		
Ca	o	Cv	o	J^	o	PD	o	PY	o	Vs	o		
Cd	o	Cw	o	NN	o	PE	x	PZ	o	Vt	x		

Test PL

PL													
A2	o	Ch	o	Cy	o	P0	o	PH	o	VB	o	RF	o
AA	345	Cj	o	Cz	o	P1	x	PJ	o	Vc	o	RR	o
AC	o	Ck	o	C}	o	P4	o	PK	o	Ve	o	RV	o
AG	345	Cl	o	Db	o	P5	x	PL	o	Vf	o	TT	o
AM	345	Cn	o	Dg	o	P6	o	PP	o	Vi	o	ZZ	o
AO	o	Co	o	II	o	P7	o	PQ	o	Vm	o	<s>	o
AU	345	Cr	o	J*	o	P8	o	PS	o	Vp	o		
C=	o	Cu	o	J,	o	P9	x	PW	o	Vq	x		
Ca	o	Cv	o	J^	o	PD	345	PY	o	Vs	o		
Cd	o	Cw	o	NN	345	PE	x	PZ	o	Vt	x		

Test PP

PP													
A2	o	Ch	o	Cy	o	P0	o	PH	o	VB	o	RF	o
AA	o	Cj	o	Cz	o	P1	o	PJ	o	Vc	o	RR	o
AC	o	Ck	o	C}	o	P4	o	PK	o	Ve	o	RV	o
AG	o	Cl	o	Db	o	P5	x	PL	o	Vf	o	TT	o
AM	o	Cn	o	Dg	o	P6	o	PP	o	Vi	o	ZZ	o
AO	o	Co	o	II	o	P7	o	PQ	o	Vm	o	<s>	o
AU	o	Cr	o	J*	o	P8	o	PS	o	Vp	o		
C=	o	Cu	o	J,	o	P9	x	PW	o	Vq	x		
Ca	o	Cv	o	J^	o	PD	o	PY	o	Vs	o		
Cd	o	Cw	o	NN	o	PE	o	PZ	o	Vt	x		

Test PQ

PQ													
A2	o	Ch	o	Cy	o	P0	o	PH	o	VB	o	RF	o
AA	o	Cj	o	Cz	o	P1	o	PJ	o	Vc	o	RR	o
AC	o	Ck	o	C}	o	P4	o	PK	o	Ve	o	RV	o
AG	o	Cl	o	Db	o	P5	x	PL	o	Vf	o	TT	o
AM	o	Cn	o	Dg	o	P6	o	PP	o	Vi	o	ZZ	o
AO	o	Co	o	II	o	P7	o	PQ	o	Vm	o	<s>	o
AU	o	Cr	o	J*	o	P8	o	PS	o	Vp	o		
C=	o	Cu	o	J,	o	P9	x	PW	o	Vq	x		
Ca	o	Cv	o	J^	o	PD	o	PY	o	Vs	o		
Cd	o	Cw	o	NN	o	PE	x	PZ	o	Vt	x		

Test PS

PS													
A2	o	Ch	345	Cy	345	P0	x	PH	o	VB	o	RF	o
AA	345	Cj	45	Cz	o	P1	x	PJ	o	Vc	o	RR	o
AC	x	Ck	45	C}	o	P4	o	PK	x	Ve	o	RV	o
AG	345	Cl	345	Db	o	P5	x	PL	x	Vf	o	TT	o
AM	345	Cn	345	Dg	o	P6	o	PP	x	Vi	o	ZZ	o
AO	x	Co	o	II	o	P7	o	PQ	x	Vm	o	<s>	o
AU	345	Cr	345	J*	o	P8	x	PS	x	Vp	o		
C=	o	Cu	o	J,	o	P9	x	PW	x	Vq	x		
Ca	5	Cv	o	J^	o	PD	o	PY	x	Vs	o		
Cd	345	Cw	45	NN	345	PE	x	PZ	x	Vt	x		

Test PW

PW													
A2	o	Ch	o	Cy	o	P0	o	PH	o	VB	o	RF	o
AA	o	Cj	o	Cz	o	P1	o	PJ	o	Vc	o	RR	o
AC	o	Ck	o	C}	o	P4	o	PK	o	Ve	o	RV	o
AG	o	Cl	o	Db	o	P5	x	PL	o	Vf	o	TT	o
AM	o	Cn	o	Dg	o	P6	o	PP	o	Vi	o	ZZ	o
AO	o	Co	o	II	o	P7	o	PQ	o	Vm	o	<s>	o
AU	o	Cr	o	J*	o	P8	o	PS	o	Vp	o		
C=	o	Cu	o	J,	o	P9	x	PW	o	Vq	x		
Ca	o	Cv	o	J^	o	PD	o	PY	o	Vs	o		
Cd	o	Cw	o	NN	o	PE	o	PZ	o	Vt	x		

Test PY

PY													
A2	o	Ch	o	Cy	o	P0	o	PH	o	VB	o	RF	o
AA	o	Cj	o	Cz	o	P1	o	PJ	o	Vc	o	RR	o
AC	o	Ck	o	C}	o	P4	o	PK	o	Ve	o	RV	o
AG	o	Cl	o	Db	o	P5	x	PL	o	Vf	o	TT	o
AM	o	Cn	o	Dg	o	P6	o	PP	o	Vi	o	ZZ	o
AO	o	Co	o	II	o	P7	o	PQ	o	Vm	o	<s>	o
AU	o	Cr	o	J*	o	P8	o	PS	o	Vp	o		
C=	o	Cu	o	J,	o	P9	x	PW	o	Vq	x		
Ca	o	Cv	o	J^	o	PD	o	PY	x	Vs	o		
Cd	o	Cw	o	NN	o	PE	o	PZ	o	Vt	x		

Test PZ

PZ													
A2	o	Ch	o	Cy	o	P0	o	PH	o	VB	o	RF	o
AA	345	Cj	o	Cz	o	P1	x	PJ	o	Vc	o	RR	o
AC	o	Ck	o	C}	o	P4	x	PK	o	Ve	o	RV	o
AG	345	Cl	o	Db	o	P5	x	PL	o	Vf	o	TT	o
AM	345	Cn	o	Dg	o	P6	o	PP	o	Vi	o	ZZ	o
AO	34	Co	o	II	o	P7	o	PQ	o	Vm	o	<s>	o
AU	345	Cr	o	J*	o	P8	o	PS	o	Vp	o		
C=	o	Cu	o	J,	o	P9	x	PW	o	Vq	x		
Ca	o	Cv	o	J^	o	PD	345	PY	o	Vs	o		
Cd	o	Cw	o	NN	345	PE	o	PZ	o	Vt	x		

A.8 V - slovesá

Modul obsahuje testy kolokácií pre každý zo slovesných poddruhov. Pri väčšine z nich nie sú známe žiadne pravidlá, ktoré by využívali bohatú morfologickú informáciu, ktorú väčšina slovies obsahuje. Výnimku tvoria poddruhy `VB` (všeobecné slovesá v prítomnom a budúcom čase) a `Vp` a `Vs` (trpné a činné príčastia). Každé zo slovies sa môže nachádzať na konci vety.

Test VB

VB													
A2	o	Ch	o	Cy	o	P0	o	PH	o	VB	o	RF	o
AA	o	Cj	o	Cz	o	P1	x	PJ	o	Vc	o	RR	o
AC	4	Ck	o	C}	o	P4	o	PK	o	Ve	o	RV	o
AG	o	Cl	o	Db	o	P5	x	PL	o	Vf	o	TT	o
AM	4	Cn	o	Dg	o	P6	o	PP	o	Vi	o	ZZ	o
AO	4	Co	o	II	o	P7	o	PQ	o	Vm	o	<s>	o
AU	o	Cr	o	J*	o	P8	o	PS	o	Vp	o		
C=	o	Cu	o	J,	o	P9	x	PW	o	Vq	x		
Ca	o	Cv	o	J^	o	PD	o	PY	o	Vs	o		
Cd	o	Cw	o	NN	o	PE	o	PZ	o	Vt	x		

Test Vc

Vc													
A2	o	Ch	o	Cy	o	P0	o	PH	o	VB	o	RF	o
AA	o	Cj	o	Cz	o	P1	x	PJ	o	Vc	x	RR	o
AC	o	Ck	o	C}	o	P4	o	PK	o	Ve	o	RV	o
AG	o	Cl	o	Db	o	P5	x	PL	o	Vf	o	TT	o
AM	o	Cn	o	Dg	o	P6	o	PP	o	Vi	o	ZZ	o
AO	o	Co	o	II	o	P7	o	PQ	o	Vm	o	<s>	o
AU	o	Cr	o	J*	x	P8	o	PS	o	Vp	o		
C=	o	Cu	o	J,	o	P9	x	PW	o	Vq	x		
Ca	o	Cv	o	J^	o	PD	o	PY	o	Vs	o		
Cd	o	Cw	o	NN	o	PE	x	PZ	o	Vt	x		

Test Ve

Ve													
A2	o	Ch	o	Cy	o	P0	o	PH	o	VB	o	RF	x
AA	o	Cj	o	Cz	o	P1	x	PJ	x	Vc	o	RR	o
AC	o	Ck	o	C}	o	P4	o	PK	o	Ve	o	RV	o
AG	o	Cl	o	Db	o	P5	x	PL	o	Vf	o	TT	o
AM	o	Cn	o	Dg	o	P6	o	PP	o	Vi	o	ZZ	o
AO	x	Co	o	II	o	P7	o	PQ	o	Vm	x	<s>	o
AU	o	Cr	o	J*	x	P8	o	PS	o	Vp	o		
C=	o	Cu	o	J,	o	P9	x	PW	o	Vq	x		
Ca	o	Cv	o	J^	o	PD	o	PY	x	Vs	o		
Cd	o	Cw	o	NN	o	PE	x	PZ	o	Vt	x		

Test Vf

Vf													
A2	o	Ch	o	Cy	o	P0	o	PH	o	VB	o	RF	o
AA	o	Cj	o	Cz	o	P1	x	PJ	o	Vc	o	RR	o
AC	o	Ck	o	C}	o	P4	o	PK	o	Ve	o	RV	o
AG	o	Cl	o	Db	o	P5	x	PL	o	Vf	o	TT	o
AM	o	Cn	o	Dg	o	P6	o	PP	o	Vi	o	ZZ	o
AO	o	Co	o	II	o	P7	o	PQ	o	Vm	o	<s>	o
AU	o	Cr	o	J*	o	P8	o	PS	o	Vp	o		
C=	o	Cu	o	J,	o	P9	x	PW	o	Vq	x		
Ca	o	Cv	o	J^	o	PD	o	PY	o	Vs	o		
Cd	o	Cw	o	NN	o	PE	o	PZ	o	Vt	x		

Test Vi

Vi													
A2	o	Ch	o	Cy	o	P0	o	PH	o	VB	o	RF	x
AA	o	Cj	o	Cz	o	P1	x	PJ	x	Vc	o	RR	o
AC	o	Ck	o	C}	o	P4	o	PK	o	Ve	o	RV	o
AG	o	Cl	o	Db	o	P5	x	PL	o	Vf	o	TT	o
AM	x	Cn	o	Dg	o	P6	o	PP	o	Vi	o	ZZ	o
AO	o	Co	o	II	o	P7	o	PQ	o	Vm	o	<s>	o
AU	o	Cr	o	J*	o	P8	o	PS	o	Vp	o		
C=	o	Cu	o	J,	o	P9	x	PW	*	Vq	x		
Ca	o	Cv	o	J^	o	PD	o	PY	o	Vs	o		
Cd	o	Cw	o	NN	o	PE	x	PZ	o	Vt	x		

* Zámeno záporné (značka PW) môže za rozkazovacím spôsobom nasledovať jedine ak sloveso je v negovanom tvare (má predponu *ne-*).

Test Vm

Vm													
A2	o	Ch	o	Cy	o	P0	o	PH	o	VB	o	RF	x
AA	o	Cj	o	Cz	o	P1	x	PJ	o	Vc	o	RR	o
AC	o	Ck	o	C}	o	P4	o	PK	o	Ve	o	RV	o
AG	o	Cl	o	Db	o	P5	x	PL	o	Vf	o	TT	o
AM	x	Cn	o	Dg	o	P6	o	PP	o	Vi	o	ZZ	o
AO	o	Co	o	II	o	P7	o	PQ	o	Vm	o	<s>	o
AU	o	Cr	o	J*	o	P8	o	PS	o	Vp	o		
C=	o	Cu	o	J,	o	P9	x	PW	*	Vq	x		
Ca	o	Cv	o	J^	o	PD	o	PY	o	Vs	o		
Cd	o	Cw	o	NN	o	PE	x	PZ	o	Vt	x		

Test Vp

Vp													
A2	o	Ch	o	Cy	o	P0	o	PH	o	VB	o	RF	o
AA	o	Cj	o	Cz	o	P1	x	PJ	o	Vc	o	RR	o
AC	4	Ck	o	C}	o	P4	o	PK	o	Ve	o	RV	o
AG	o	Cl	o	Db	o	P5	x	PL	o	Vf	o	TT	o
AM	4	Cn	o	Dg	o	P6	o	PP	o	Vi	o	ZZ	o
AO	4	Co	o	II	o	P7	o	PQ	o	Vm	o	<s>	o
AU	o	Cr	o	J*	o	P8	o	PS	o	Vp	o		
C=	o	Cu	o	J,	o	P9	x	PW	o	Vq	x		
Ca	o	Cv	o	J^	o	PD	o	PY	o	Vs	o		
Cd	o	Cw	o	NN	o	PE	o	PZ	o	Vt	x		

Test Vs

Vs													
A2	o	Ch	o	Cy	o	P0	x	PH	o	VB	o	RF	o
AA	o	Cj	o	Cz	o	P1	x	PJ	o	Vc	o	RR	o
AC	4	Ck	o	C}	o	P4	o	PK	o	Ve	o	RV	o
AG	o	Cl	o	Db	o	P5	x	PL	o	Vf	o	TT	o
AM	4	Cn	o	Dg	o	P6	o	PP	o	Vi	o	ZZ	o
AO	4	Co	o	II	o	P7	o	PQ	o	Vm	o	<s>	o
AU	o	Cr	o	J*	o	P8	o	PS	o	Vp	o		
C=	o	Cu	o	J,	o	P9	x	PW	o	Vq	x		
Ca	o	Cv	o	J^	o	PD	o	PY	o	Vs	o		
Cd	o	Cw	o	NN	o	PE	x	PZ	o	Vt	x		

Dodatok B

Obsah CD-ROM

Súčasťou práce je CD-ROM s elektronickou verziou tejto práce vo formáte PDF a aplikáciou SyMorAn, vrátane jej zdrojových kódov. V tomto dodatku popisujeme adresárovú štruktúru CD-ROMu.

```
doc/                obsahuje elektronickú podobu tejto práce
                    vo formáte PDF a dokument [4]
README.txt          obsahuje základné infomácie o obsahu CD-ROMu
                    vrátane tohto popisu adresárovej štruktúry
symoran/            obsahuje aplikáciu SyMorAn
   bin/             obsahuje binárku aplikácie SyMorAn
   doc/             obsahuje dokumentáciu k aplikácii
      README.txt    nápoveda k aplikácii
      doxygen/      programátorská dokumentácia aplikácie v podobe
                    HTML stránok vygenerová systémom Doxygen
   src/             zdrojové kódy aplikácie vrátane Makefile  súboru
```